Gestural Annotation of User Interface Requirements in Crowd Elicitation

ANDREW ARNITA, Université catholique de Louvain, Belgium
EMILIO INSFRAN, Universitat Politècnica de València, Spain
NUWAN T. ATTYGALLE and JEAN VANDERDONCKT, Université catholique de Louvain, Belgium

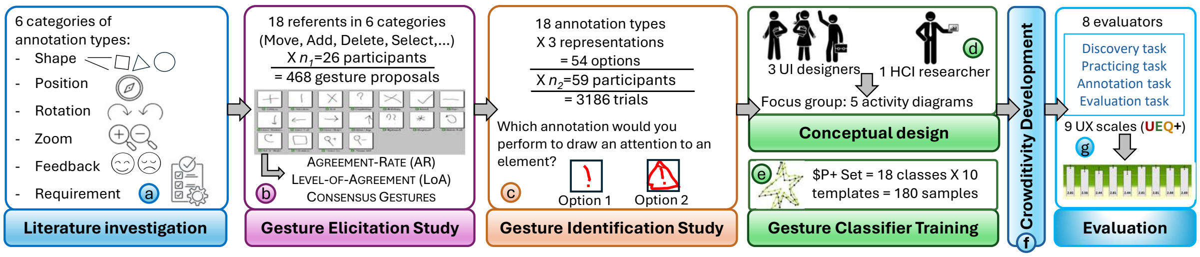


Fig. 1. Pipeline of the development process: a literature investigation yielded 6 categories of annotation types (a), feeding a gesture elicitation study of 18 annotations (b), confirmed by an identification study (c), whose results are used to train a 2D multi-stroke gesture classifier (d) in Crowditivity. In parallel, a focus group designed and agreed on five activity diagrams (e) used to develop Crowditivity (f), that is further evaluated in the last step for its user experience (g).

User interface requirements elicitation is an early and critical stage in the software development life cycle: requirements are elicited from stakeholders, and user interface elements that should satisfy these requirements are also created, shared, and edited. To effectively capture the diverse perspectives of stakeholders on these requirements, crowd elicitation is a promising approach to leverage the collective knowledge of a time- and space-distributed group of stakeholders to manage user interface requirements and their associated user interface elements. Annotating these elements manually in freeform digital ink leads to results that are difficult to interpret and time-consuming to exploit by designers and developers. Instead, we propose that stakeholders annotate user interface elements by commonly-agreed gestures that will be automatically recognized to populate 10 annotation metrics. To this end, we present Crowditivity, a publicly available web application that effectively harnesses crowd elicitation in three related activities: (1) collecting and managing user interface functional and non-functional requirements according to an adapted Volere Shell template; (2) assigning user interface elements of a mockup or prototype to requirements to satisfy them; and (3) enabling stakeholders to annotate these elements by ink-based gestures. A 2D multi-stroke gesture classifier is specifically tailored to recognize gesture classes corresponding to 18 annotation types, that are empirically validated through a gesture elicitation study (n_1=26) confirmed by an identification study (n_2=59). Through an evaluation of a case study for a web application, we investigated how eight representative designers valued the Crowditivity platform. We finally define and illustrate a set of ten metrics to quantify the output of Crowditivity in terms of annotations and annotators resulting from the gesture recognition.

Authors' Contact Information: Andrew Arnita, andrew_arnita@hotmail.com, Université catholique de Louvain, Louvain Research Institute in Management and Organizations (LouRIM), Louvain-la-Neuve, Belgium; Emilio Insfran, Universitat Politècnica de València, Departamento de Sistemas Informáticos y Computación, Valencia, Spain, einsfran@dsic.upv.es; Nuwan T. Attygalle, nuwan.attygalle@uclouvain.be; Jean Vanderdonckt, jean.vanderdonckt@uclouvain.be, Université catholique de Louvain, Louvain Research Institute in Management and Organizations (LouRIM), Louvain-la-Neuve, Belgium.

ACM 2573-0142/2026/6-ARTEICS017
https://doi.org/10.1145/3816769

CCS Concepts: • **Software and its engineering → Requirements analysis**; • **Hardware → Touch screens**; • **Human-centered computing → Gestural input**; *User studies*; Laboratory experiments; *Participatory design.*

Additional Key Words and Phrases: Annotation, Crowd elicitation, Mockups, Prototypes, Requirements elicitation, Requirements engineering, Stroke gesture recognition.

ACM Reference Format:
Andrew Arnita, Emilio Insfran, Nuwan T. Attygalle, and Jean Vanderdonckt. 2026. Gestural Annotation of User Interface Requirements in Crowd Elicitation. *Proc. ACM Hum.-Comput. Interact.* 10, 4, Article EICS017 (June 2026), 40 pages. https://doi.org/10.1145/3816769

1 Introduction

In the field of Requirement Engineering (RE) [30,42], requirements elicitation initiates the software development process by uncovering and articulating stakeholders' needs, constraints, and expectations before design and implementation begin [8] in form of requirements [52]. Requirements elicitation is widely recognized as a foundational and decisive stage in the software development life cycle [26], particularly in the engineering of interactive systems and their User Interfaces (UI) [36] (see Maguire and Bevan [46] for a review). In this work, we adopt a precise definition of *Requirements Elicitation* as the acquisition of information from an individual or group of end users, which we refer to as stakeholders, in a manner that does not explicitly disclose the intent of the inquiry. More broadly, elicitation encompasses any technique used to discreetly extract information that is not readily available but is essential for a specific development purpose [26]. In the context of UI engineering, elicitation not only gathers requirements from stakeholders but also informs the creation and sharing of interface elements intended to satisfy those requirements.

Despite its centrality, UI requirements elicitation often relies on manual annotation technologies [50] of UI design artifacts [39], such as textual or multimedia descriptions [22], structured forms, images [28], facilitator-led documentation resulting from scenarios [3] or interviews [11,49], that can be tedious to produce and challenging to interpret [18]. These approaches frequently struggle to capture dynamic interaction behaviors [16], navigation flows, and nuanced user intentions [25]. As a result, ambiguity persists between technical and non-technical stakeholders, potentially propagating misalignment into later UI design and prototyping stages.

To address these limitations, we investigate the use of gesture-based annotation systems [63] for UI requirements elicitation [36], acknowledging the critical role of annotations in user-centered design [24]. An *annotation system* in this context refers to a structured framework that enables stakeholders to augment UI artifacts, such as sketches [4,58], wireframes [35], empathy maps [31] with personas [32], storyboards [23], mockups and prototypes [25] with additional semantic information. Rather than relying solely on text, our system allows stakeholders to use stroke gestures [5] (e.g., arrows indicating navigation, circles highlighting components, symbols representing user actions [16]) to visually and directly express their intentions. These gestures serve as expressive, embodied annotations that are digitally captured and linked to UI elements associated to requirements.

The gestural approach offers several advantages. First, it enhances clarity and reduces ambiguity by allowing stakeholders to communicate through intuitive visual marks [63]. Second, it lowers the cognitive burden associated with rigid documentation formats [75], encouraging broader participation of end users [51], which can be considered sometimes as software developers [15]. Third, it supports agile [34] collaboration by enabling participants to sketch ideas directly onto shared UI artifacts [59]. Most importantly, gestural annotations can be automatically recognized, systematically interpreted, and translated into structured comments on UI artifacts associated to requirements [64]. Recognized gestures can be categorized and associated with UI behaviors, interaction types, or priority levels. This integration not only improves the multi-fidelity [16] of elicited requirements but also enables traceability, consistency, and automated analysis throughout subsequent development stages.

The value of annotation extends beyond elicitation into prototyping, where annotating UI artifacts enables stakeholders to align their expectations with evolving UI design alternatives [24,28]. By supporting gesture recognition, our approach transforms annotations from informal marks into analyzable, quantifiable data. For example, gesture frequency may indicate interaction priority, patterns of navigation arrows may reveal user flow complexity, and clustering of highlights may signal focal interface elements. These data provide early insights into user expectations and design trade-offs, strengthening decision-making in the formative phases of UI engineering. This work situates gestural annotation within *crowd elicitation* [1], the practice of gathering requirements from a distributed [45] and diverse group of participants rather than a limited set of stakeholders [28]. Crowd elicitation leverages collective intelligence to surface latent needs, edge cases, and alternative perspectives that may not emerge in traditional sessions [47]. However, scaling elicitation to the crowd introduces challenges of consistency, interpretation, and agreement [67] among the answers returned by members of the crowd. Our approach addresses these challenges by designing user-defined gestures [73] instead of system-defined [74] for annotating UI requirements. More specifically, this paper contributes:

- An empirical understanding and formalization of 18 annotation types of UI artifacts associated to UI requirements, organized into 6 conceptual categories, grounded in two complementary studies:
  (1) A gesture elicitation study (n_1=26) exploring how participants represent annotation types by gestures.
  (2) A gesture identification study (n_2=59) evaluating recognition agreement and interpretability.
- A curated gesture dataset representing the 18 annotation types that maximize inter-participant agreement, designed explicitly for use in crowd-based annotation environments.
- Crowditivity, a publicly available web application that effectively harnesses crowd elicitation in three related activities: managing UI requirements according to an adapted Volere Shell template; assigning these requirements to UI artifacts, and enabling stakeholders to annotate them by stroke gestures.
- A 2D multi-stroke gesture classifier specifically tailored to recognize 18 classes of the curated gesture dataset.
- An evaluation of a case study for a web application revealing how eight designers valued Crowditivity.
- A set of ten metrics to quantify the output of Crowditivity in terms of annotations and annotators resulting from the gesture recognition.

By integrating intuitive gestural annotation of UI requirements in crowd elicitation, with automated gesture recognition on empirically validated annotation categories, this work advances a scalable approach to UI requirements engineering. Gestural annotation in crowd elicitation is expected not only to enhances expressiveness and clarity but also establishes a structured foundation for analyzing, prioritizing, and tracing UI requirements throughout the development life cycle of interactive systems.

To this end, the remainder of this paper is structured as follows: Section 2 reviews work related to requirements engineering and elicitation with a focus on gestural annotations of UI requirements. Section 3 conducts a crowd-based gesture elicitation study to determine the preferred gesture representations of 18 annotation types conceptualized in 6 categories. To reach a commonly-agreed representation [69] of these annotations, which is key for crowdsourcing [28], Section 4 conducts a crowd-based gesture identification study to validate the two/three most preferred gesture representations that will be used to train a 2D multi-stroke gesture recognizer to be embedded in Crowditivity. Informed by 6 activity diagrams, Section 5 delivers a Crowditivity system walkthrough. The system is evaluated on a case study for a web application by performing a UEQ+ evaluation with 8 representative UI designers in Section 6. Section 7 defines and illustrates 10 metrics that describe the behavior of annotators as captured by Crowditivity. Section 8 discusses the limitations of Crowditivity and the threats to validity encountered in the user studies. Section 9 concludes this paper and presents some avenues for this work.

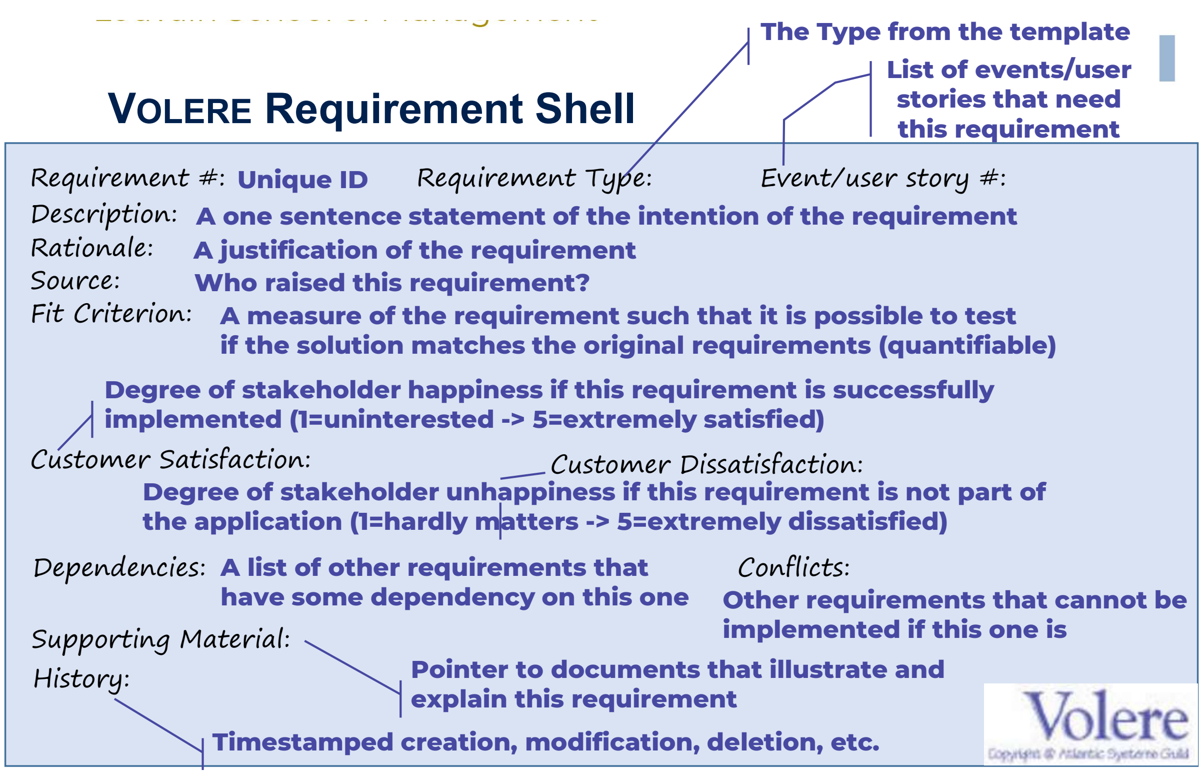


Fig. 2. VOLERE Shell Template used for capturing user interface requirements.

Fig. 1 shows the pipeline of the development process followed in this work. First, a literature investigation (Fig. 1-a) enabled us to define and identify six categories of annotation types (Section 2): shape, position, rotation, zoom, feedback, and requirement. These categories were subsequently used as referents for a gesture elicitation study (Fig. 1-b) of 18 individual annotations (Section 3). The 3 most frequently preferred representations of the 18 annotation types extracted from this study initiated a gesture identification study (Fig. 1-c) in Section 4, whose results are used to train a 2D multi-stroke gesture classifier (Fig. 1-e). This classifier was then integrated in CROWDITIVITY, a web application that was conceptually designed (Fig. 1-d) and developed (Fig. 1-f) to harness crowd elicitation of user interface requirements, that is further evaluated for its user experience (Fig. 1-g).

2 Related Work

When engineering an interactive application, a heterogeneous set of stakeholders, such as end users and their representatives, designers, developers, and business representatives, contribute requirements that must be systematically articulated, documented, managed, and validated [42]. The diversity of stakeholder perspectives necessitates the use of shared representational conventions to ensure consistency and mutual understanding. To this end, several guidelines, standards, and templates have been established to formalize requirements specification, including the System Requirements Specification [19], System Requirements Template, and the ISO/IEC/IEEE 29148:2018 standard. These frameworks emphasize the importance of adopting a common language and structure for describing requirements to support traceability, verifiability, and life cycle management.

Methodological approaches such as Volere [52] further refine this structuring by linking stakeholder needs to both functional and non-functional requirements. Functional requirements describe the system's observable behaviors and services (e.g., enabling a user to download a digital resource), whereas non-functional requirements define quality attributes such as performance, usability, security, or reliability (e.g., ensuring that the download operation completes within an acceptable time frame). The Volere Shell template (Fig. 2) provides a structured mechanism for capturing requirements in a manner that facilitates testing, traceability, and progressive refinement toward implementation.

In this work, we define a *User Interface requirement* as a specific, measurable, and verifiable specification describing how the UI of an interactive application should appear, behave, or operate to satisfy stakeholder needs. UI requirements encompass presentation aspects (e.g., visual layout and styling), behavioral aspects (e.g., interaction flows and state transitions), and operational aspects (e.g., system responses to user input). They specify the UI widgets, interaction patterns, interaction modalities, and techniques necessary to achieve the intended user experience.

UI requirements typically address properties related to user experience [11], such as visual organization, navigation structure, responsiveness, accessibility, usability criteria, and the integration of UI components. To be effective, they must be expressed with sufficient clarity to guide designers and end users conceptually, while remaining precise and actionable for developers and testers during implementation and validation.

Furthermore, UI requirements are characterized as either *intentional* or *extensional* [13]. Intentional requirements articulate general principles, desired properties, or high-level design objectives. Extensional requirements, in contrast, are grounded in specific instances [6], concrete scenarios [3], predefined examples and user stories [47], storyboards [23], or annotated mockups [24,25]. This distinction is particularly relevant in crowd-based design contexts, where requirements may emerge either from abstract expectations or from direct interaction with concrete design artifacts.

2.1 Requirements Engineering and Elicitation

Pekkola et al. [51] argue that "different communication means for gathering and elucidating requirements in a workplace would produce better systems from the end-users' point of view." Communication with end-users during the development process is key to understanding their needs.

There are several tools for supporting requirements engineering and requirements elicitation in general, but they are not targeted to UI requirements. ReqView is a requirements management tool enabling stakeholders to manage Volere requirements throughout the development life cycle structured according to a V-model, where each step should be verified and validated. While ReqView provides templates for various aspects of the software, such as the technical constraints or the business requirements, it does not offer any template regarding UI aspects, which is an intrinsic drawback for interactive applications. In addition, this software runs locally, therefore not appropriate for crowd elicitation. Storywise is specialized in the efficient and AI-assisted creation and initial structuring of requirements, such as user stories, tasks model, acceptance criteria, from diverse, often unstructured inputs.

Sucre (Scenario and Use-Case based Requirements Engineering) [3] consists of a software framework based on a UI model (UCM-UI), a UI model enriching Use Case Maps (UCMs) with a visual notation for modeling and specifying the UI requirements. Scenarios and use cases are the means to capture and represent the UI requirements to be communicated to the project stakeholders, mainly the end-users but not only. The consistency, completeness, and precision of the UCM-UI model are validated using heuristics. Sucre also defines metrics to predict usability from scenarios and use cases, such as simple structural measures, content-sensitive, and task-sensitive metrics. To gracefully evolve from informal requirements to more formal ones, Sucre starts from the semi-formal requirements expressed in UCM-UI and ends up with formal requirements expressed in UML and

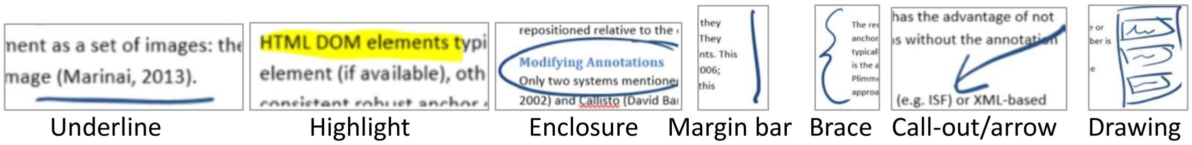


Fig. 3. Classification of annotation types for an electronic document [63].

LOTOS specifications. The USE software [75] provides UI elements from the user/task model and examines the usability of the designed UI through the storyboarding mechanism. WebStruine [59] also enables stakeholders to capture UI requirements specifically while RedSeeds [78] is not restricted to UI requirements. While this facility is important, it does not enable the stakeholders to create UI elements and to annotate them, as does Panda [25] to decide the best UI alternatives and options finally.

In sum, existing software supports managing requirements in general or UI requirements in particular, but without linking them explicitly, therefore leaving intentional UI requirements aside.

2.2 Elicitation for User Interfaces

Crowdlicit [1] is a software for conducting distributed end-user elicitation and identification studies [2] to be performed online using a crowd of participants and produce quality results in a very short amount of time. To elicit UI requirements on actions such as voice commands, icons, button labels, menu items, and gestures, the process is decomposed into four stages: (1) create a labeled study with a description explaining the purpose of that study, (2) choose referents corresponding to an action to be executed, (3) select a symbol type specifying how to invoke the referents (e.g., "please open/close this file"), (4) fill out user-intended questions, and (5) analyze its results. CrowdSensus [2] is a crowd-based tool enabling researchers to crowdsource the similarity judgments central to agreement analysis as provided by Crowdlicit using five automatic clustering algorithms to analyze the requirements elicited from end-users: steepest-ascent hill climbing, shotgun hill climbing, simulated annealing, a genetic algorithm, and correlation clustering. CrowdSensus relies on Amazon Mechanical Turk (AMT) to send the study to a crowd of participants.

Gelicit [45] is an online application for gesture elicitation in which stakeholders can enter gesture proposals for future commands of the system using different sensors. Although this software limits the types of gestures, such as stroke gestures and webcam-based mid-air gestures, it enables stakeholders to perform the elicitation in a distributed way. GestureWiz [61] does a similar job for gesture elicitation studies [72]. Hesenius et al. [27] performed a gesture elicitation study about expressing spreadsheets operations through sketching to observe that many syntactical variations emerge for gestures elicited for the same command. For example, to select a cell or a range of cells, participants proposed gestures such as a strike, an underline, an enclosure, a squiggly line, a "X" mark, and a gross point. While these gestures are aimed at executing the corresponding command, their representation can be considered as another set of annotations for spreadsheets. In a follow-up to this elicitation study, Leson et al. [38] found no difference in execution time between commands given verbally and gesturally, but noted that this depends primarily on the user's native language. SnappView [18] supports the UI prototyping stage by enabling stakeholders to submit UI elements and to freely annotate them, but these annotations are not recognized and the fragments are not linked to requirements.

In sum, the elicitation currently supported by the tools seems limited in terms of coverage (a single step is generally covered, but no more, without continuity), requirements (mostly expressed implicitly, not explicitly), symbols manipulated (mainly elicited gestures), links between the various concepts useful for elicitation. In addition, the annotation stage is blended into a more general process, without being treated as an integral part of it. That is why we devote the following sub-section to it.

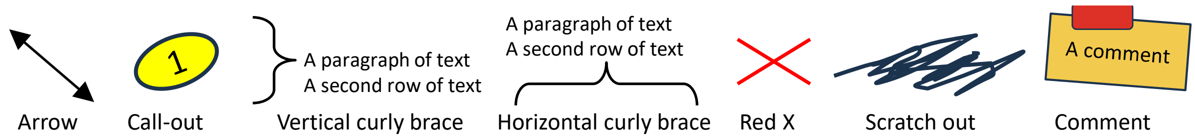


Fig. 4. Annotation types of Balsamiq.

2.3 User Interface Annotation Types

In UI engineering, an *annotation* is a mark made on a mockup or prototype to give instructions about specific UI components that need modification. For example, the term annotation comprises several methods in situations involving text, including underlining, circling, striking out, highlighting text, and writing additional comments in the margin [29]. For example, Sutherland et al. [63] analyzed a large number of references related to inking of electronic documents to identify seven annotation types (Fig. 3): underline, highlight, enclosure, margin bar, brace, call-out/arrow, and drawing.

These activities stimulate critical thinking in a process called *active reading* [62], a reading activity determined to understand and evaluate the mockup for its relevance to the end user's needs. In addition, annotations serve as a communication technique between designers, developers, and other stakeholders working on software development. An important role for annotations is to act as signallings for important information [18,25], and providing such information helps ensure that the software meets the UI requirements and that all stakeholders have a clear understanding of its design, no matter the form or purpose of the annotations.

Ovsiannikov et al. [50] discuss software annotation technology and the design of annotation. After reviewing 21 annotation software, Li et al. [39] classify annotation types into six major categories in terms of the target audience (human or computer), the rendering system (static or dynamic), the used media (physical or digital), the usage and functions of the annotation (semantic or procedural), the location of the annotation storage (in-line or stand-off) and the representation used for the annotation (freestyle or structured). According to this classification, Crowditivity envisions enriching UI requirements with annotation types targeted at both human and computer (for automatic recognition), semantic and procedural, in-line (since represented within the application), structured and freestyle (depending on the annotator prefers to rely on the gesture vocabulary for annotation or not). While these papers are useful categorizing annotation types, none of them address annotation types in UI development.

2.4 Commercial Annotation Tools

Balsamiq offers a single annotation type, i.e., a sticky textual note (Fig. 4), that can be drawn with five stencils called *callout*, *pointer* (e.g., an arrow), *bracket*, *speech bubble*, and *thought bubble*, to point towards actionable UI elements of the mockup that deserve some attention. These annotations are typically added as notes, callouts, or overlays in design tools like Figma, Adobe XD, or Sketch, or included in detailed design documentation. They ensure clarity and alignment across teams during the development process.

By reviewing related work, we observe that the main annotation types found in commercial annotation tools and research efforts are diverse: functional annotations (e.g., "the functionality of this push button is not clear"), interaction annotations (e.g., "how to reorder items in this list"), visual annotations (e.g., the padding of this widget is too high), behavioral annotations (e.g., "why this dropdown menu appears with a fade-in animation so fast?"), accessibility annotations (e.g., "the text contrast ratio does not meet WCAG AA standards"), content annotations (e.g., "this error message is not concise"), conditional annotations (e.g., "Why displaying an error message if the form is submitted ?"), linkage annotations (e.g., "Clicking 'Next' does navigate to the Payment screen"), and responsive design annotations (e.g., "This sidebar should be hidden on mobile devices". There is no precise existing classification of annotation types used in UI development and even less any understanding of how end users could convey these annotations by gesture.

In contrast to the aforementioned efforts, Crowditivity jointly supports the UI requirements elicitation stage and the UI early design stage by prototyping and to link them to preserve these contents for the rest of the development life cycle. None of the above tools automatically recognizes the annotations. Indeed, creating and iterating on UI mockups is a crucial but time-consuming stage in the development life cycle. Gesture recognition streamlines this process by allowing stakeholders to annotate mockups intuitively through simple gestures, which are then translated and classified into annotations or UI elements. This should reduce reliance on manual tools, preserves creative flow, and enhances collaboration by providing a universal, easy-to-adopt gesture language.

2.5 Why Gestural Annotation of User Interface Requirements

Using 2D stroke gestures [77] to annotate UI requirements can be justified from both a user experience and a requirements engineering-efficiency perspective.

First, 2D stroke gestures provide a natural and intuitive mode of annotating requirements and associated mock-ups. Stakeholder are already accustomed to drawing, circling, underlining, and sketching on paper mock-ups and prototypes. Translating this behavior into an interactive application reduces the cognitive load required to learn new tools. Instead of navigating complex menus and selecting menu items, stakeholders may associate stroke gestures to these commands [5,27] to simply “mark up” mock-ups in a way that feels immediate and familiar.

Second, stroke gestures enable faster and more expressive communication [14]. Traditional requirement annotation methods, such as written comments or structured forms, can be time-consuming and do not capture spatial intent. Stroke gestures instantly convey meaning [37] while preserving the spatial relationship between UI elements and the feedback itself, which is critical where requirements and mock-ups positioning matter. Studies [41,65] demonstrate that informal, stroke-based input supports faster and more effective communication of stakeholders’ reactions compared to comments expressed in other modalities [7], such as textual or vocal [38].

Third, stroke-based annotation supports iterative and collaborative requirements elicitation [55], especially when the requirements are elaborated in-context [54]. In early-stage of the life cycle, UI requirements may evolve rapidly. Stroke gestures allow users to annotate directly [28] on mock-ups, wireframes, or prototypes without interrupting the flow of participatory design [17]. In collaborative settings, multiple stakeholders can annotate simultaneously or asynchronously. Stroke gestures can be efficiently recognized by gesture classifiers [43].

Finally, stroke gestures enhance traceability and context retention, which is vital for validating the requirements elicited [33]. Because annotations are directly overlaid on the UI artifacts they refer to, the original context of each requirement is preserved [54]. This makes it easier for developers and designers to understand the rationale behind changes and reduces ambiguity compared to detached textual descriptions.

3 A Gesture Elicitation Study for Annotation Types

Based on a literature investigation (Fig. 1-a) on annotation types in various domains [63], including engineering interactive systems [1,18,25,29,39,53] (see Section 2.3), we classify annotation types into six major categories:

(1) *Shape*: This group includes annotations related to the shapes drawn on an image. Annotation types such as "Line" are used as basic shapes, while "Add" and "Delete" are used to add and remove elements, respectively. "Select" is used to select, designate, or identify an element.
(2) *Position*: This group includes annotations related to moving an element in an image. "Move left," "Move right," "Move down," and "Move up" are typically used to move an element in any direction.
(3) *Rotation*: This group includes annotations related to rotating and flipping elements in an image. "Rotate left" and "Rotate right" are used to rotate an element counterclockwise and clockwise, respectively.

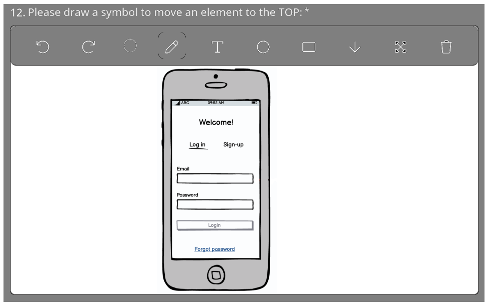


Fig. 5. Canvas for gesture proposal input.

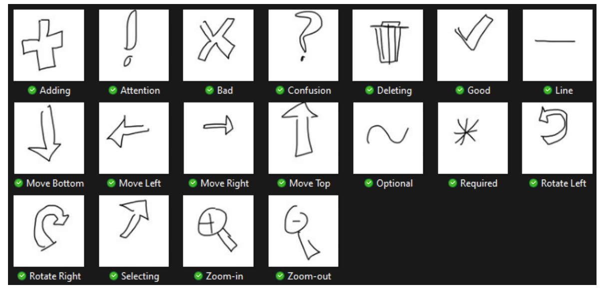


Fig. 6. Gesture proposals of a participant.

(4) *Zoom*: This group includes annotations related to the size of an element. "Enlarge" or "Zoom in" and "Shrink"/"Zoom out" are used to increase or decrease the size of an element, respectively.
(5) *Feedback*: This group includes annotations related to the feedback given by the users regarding a specific element. "Confused" is used when a participant is uncertain about what a specific element is used for, while "Good" and "Bad" are used to indicate whether the element is correctly designed or not. "Attention" indicates that the element in the mockup is important and requires some attention.
(6) *Requirement*: This group includes annotations related to the existence of an element in an image. "Required" indicates that an element is mandatory, while "Optional" is used to indicate that it is not.

Since no consensus existed on suitable representations for these annotation types [63] and since participants usually come up with many syntactical variations [27], we need to design a user-defined gesture vocabulary [74] tailored to UI annotation types. Therefore, we conducted a Gesture Elicitation Study (GES) [72,73], a participatory design technique [51] in which participants are instructed to propose a gesture in response to a referent expressing a future command (here, an annotation) of the system (Fig. 1-b) (see [70] for a review).

3.1 User Study

3.1.1 Stimuli. We defined a set of 18 *referents* [72] representative of frequent annotation types executed during UI requirements elicitation based on the six categories: line, move left/right/up/down, add, delete, select, required/optional, good/bad, confused, attention, enlarge/zoom in, shrink/zoom out (Table 1).

3.1.2 Apparatus. We developed a Form crowd-based application that captures the participant's gestures in a way that is distributed in time and space, inspired by CrowdLicit [1] and Gelicit [45], a primordial feature of crowd-based approaches. Fig. 5 reproduces a screenshot where a simple instruction is prompted (e.g., "Please draw a symbol to select an element") to avoid visual priming [48] and making any reference to any representation or look & feel. We developed this application specifically for the acquisition of stroke gestures, so that each participant can acquire each proposal in his or her own work environment, but also with the aim of retaining the gestures insofar as the corresponding class will be retained later for training. This acquisition of gestures in an environment familiar to the participant will be representative of gestures intended to train the recognizer. Usually, gesture acquisitions in elicitation and training are dissociated. Here, we combine them to cover as many natural gesture articulations as possible.

3.1.3 Procedure. We used a within-subject design where participants were elicited for one gesture per referent. After signing a GDPR-compliant consent form, participants completed a sociodemographic questionnaire, providing information about their age, gender, and handedness, and their use of computer technology. Subsequently, the participants were instructed to propose suitable gestures to annotations, i.e., gestures that fit referents well, are easy to produce and remember, and remain as natural as possible. The order of the referents was randomized.

Table 1. The 18 referents selected for annotation types in Crowditivity.

Annotation type	**Description**	**Category**
Line	Annotate a UI element by underlining, strikethrough,...	Shape
Move left	Annotate a UI element to be moved to its left	Position
Move right	Annotate a UI element to be moved to its right	Position
Move up	Annotate a UI element to be moved up, e.g., to the top of the mockup	Position
Move down	Annotate a UI element to be moved down, e.g., to the bottom of the mockup	Position
Add	Annotate a UI element to be added	Shape
Delete	Annotate a UI element to be removed from the mockup	Shape
Select	Annotate a UI element to designate it for further comments	Shape
Required	Annotate a UI element to mark its mandatory character	Requirement
Optional	Annotate a UI element to mark its facultative character	Requirement
Good	Annotate a UI element that is adequately designed	Feedback
Bad	Annotate a UI element that is not appropriately designed, to be improved	Feedback
Confused	Annotate a UI element considered unclear	Feedback
Attention	Annotate a UI element to draw attention on it	Feedback
Zoom in	Annotate a UI element to request enlargement	Zoom
Zoom out	Annotate a UI element to request shrinking	Zoom

3.1.4 Participants. Twenty-six volunteers (n_1=26, 12 women and 14 men), aged between 18 and 57 years old (*M*=28.77, *SD*=9.41, *Mdn*=25.5), were conveniently recruited via contact lists in different organizations. Their occupations included people working in administration (e.g., secretary, clerk) and social sciences (e.g., psychology, law, communication, economics, and management). All participants reported very frequent use of computers (*M*=6.81, *SD*=0.56, *Mdn*=7 on a 7-point Likert scale [40] with items ranging from 1=strongly disagree to 7=strongly agree) and smartphones (*M*=6.73, *SD*=0.71, *Mdn*=7), less frequent use of tablets (*M*=3.27, *SD*=2.30, *Mdn*=2.50), no dexterity impairments, and had normal or corrected-to-normal vision.

3.2 Results and Discussion of the Elicitation Study

We collected 26 participants × 18 annotation types = 468 gesture proposals (see Fig. 6 for gestures proposed by the participant *P*9), which were clustered into groups by similarity [9] (Table 2), inspired by the inverse dissimilarity [67]. For example, Fig. 7 shows how this clustering (Fig. 8) is applied to participant *P*2.

We then computed the Agreement-Rate (AR) [69], a measure of the level of consensus between the proposed gestures that produces values in the $[0, 1]$ interval. We computed ARs with AGATe [69] and interpreted their magnitudes according to thresholds of 0.10 (low), 0.30 (medium), 0.50 (high), and above (very high). Fig. 9 shows the distribution of the agreement rates in decreasing order of their magnitude for the 18 annotation types. Overall, ARs ranged from a maximum of 0.99 for "Line" to a minimum of 0.09 for "Optional" with an average level of agreement of 0.47 (SD=0.23, Mdn=0.5), representing high agreement according to our interpretation rules for ARs. The "Line" referent obtained the maximum rate since it already conveys the shape of the representation, even if participants produced different types of lines, some dashed, some ragged. The average rate is high, especially compared to the rates found in other GES, which are usually below 0.02 [71] (e.g., Gheran et al. [21] reported an average agreement rate of 0.112 for smart ring gestures to control devices and Zaiti et al. [76] reported 0.158 for mid-air gestures acquired with the Leap Motion controller for TV control), thereby suggesting that participants truly agreed among themselves about their proposals.

Referent	Classification Groups
Add	1, 2, 3, 4, 5
Attention	7, 8, 9, 10, 13, 14
Bad	15, 16, 17, 19, 20, 21
Confusion	18, 24, 25, 26
Delete	16, 27, 29
Good	32, 33, 34, 35, 36
Line	28
Move Bottom	37, 39
Move Left	31, 40, 41
Move Right	42, 43
Move Top	44, 46
Optional	22, 23, 48, 49, 50, 51, 53, 54, 55, 56, 57
Required	11, 58, 59, 60
Rotate Left	38, 63
Rotate Right	45, 62
Select	30, 52, 65, 66
Zoom in	47, 67, 68, 69
Zoom out	12, 70, 71, 72, 73, 74

Table 2. Assignment of gestures to classification groups [9].

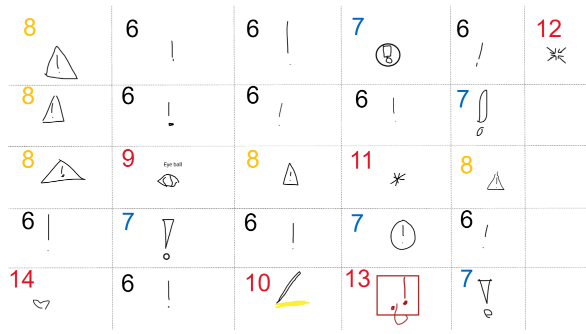


Fig. 7. A classification of gestures proposed by a participant.

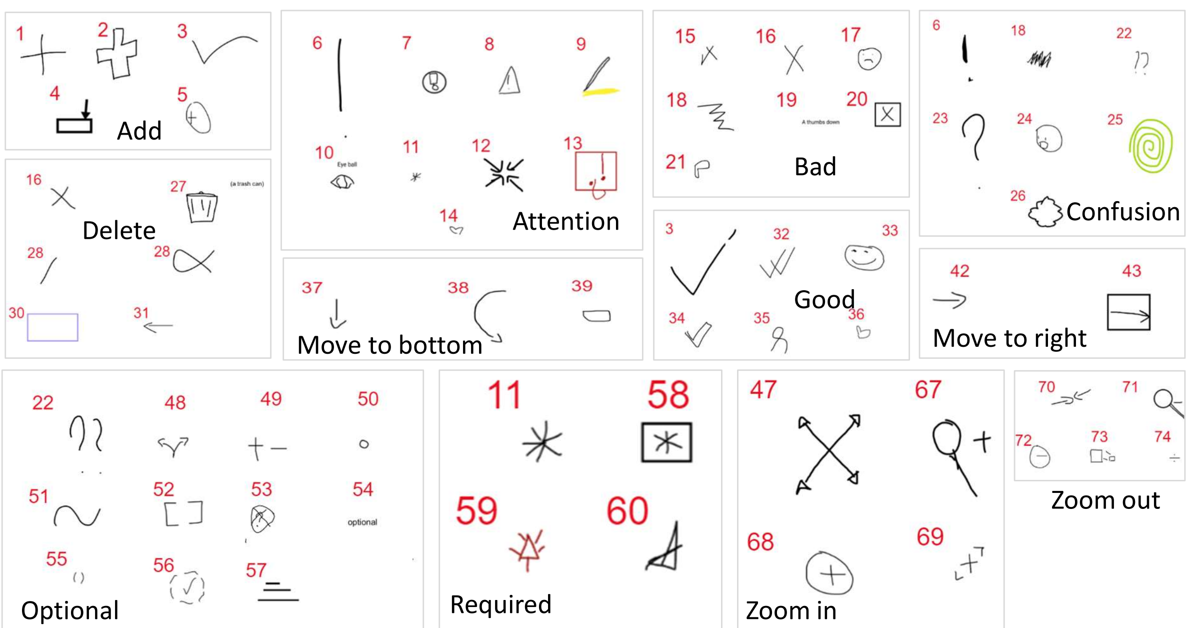


Fig. 8. Examples of templates by classification group.

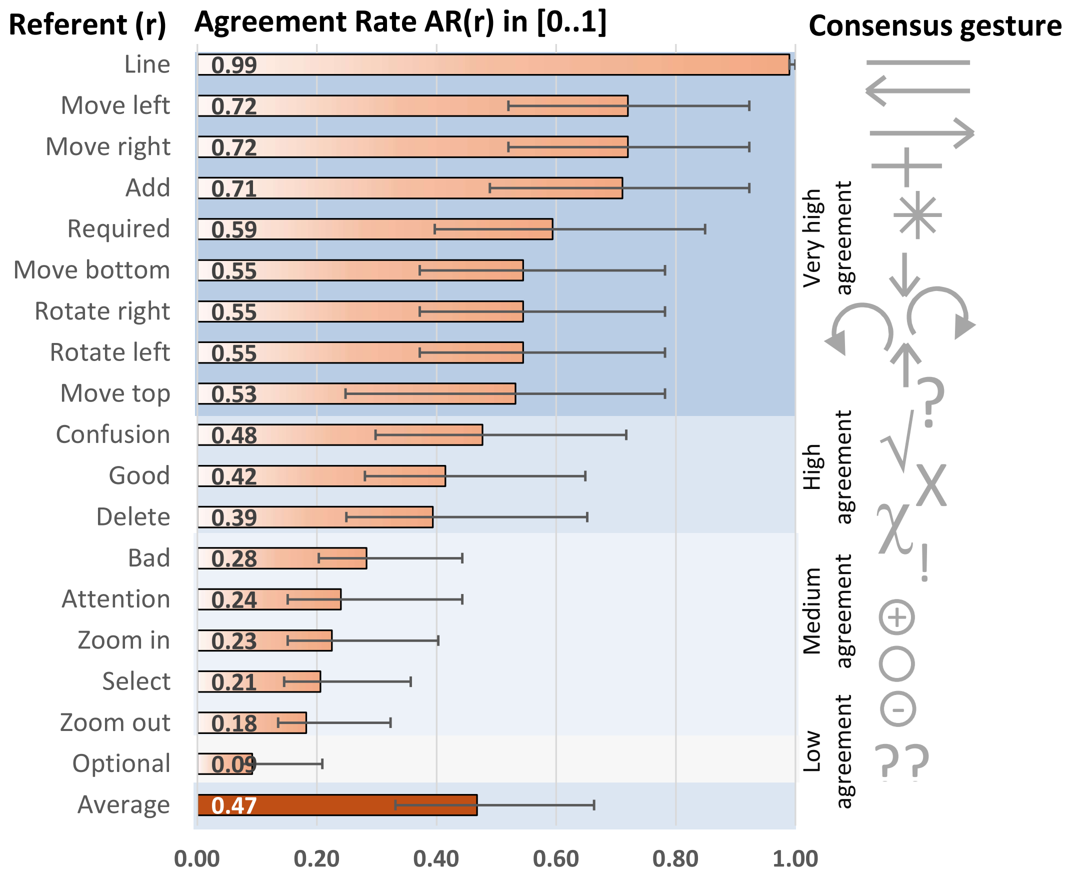


Fig. 9. Stroke gestures elicited for our annotation types sorted in decreasing order of their agreement rate $AR(r)$.

We determined 5 ranges for the Level-of-Agreement: *very high* for a rate higher than 0.5 (9/18=50% of the referents), ranging from "Line" to "Move to top", *high* for a rate between 0.3 and 0.5 (3/18=17%), such as "Confusion", "Good", and "Delete", *medium* for a rate between 0.1 and 0.3 (5/18=28%), such as "Attention", and "Zoom out", and *low* for a rate less than 0.1 (1/18=5%). Every referent above the average is considered outstanding compared to the others.

We also computed the co-agreement rate [69] to investigate whether gestures proposed for related referents were also in agreement with each other: *CR* ("Rotate left", "Rotate right")=0.545, *CR* ("Move to top", "Move to bottom", "Move to left", "Move to right")=0.529, *CR* ("Add", "Delete")=0.335, *CR* ("Good","Bad")=0.191, *CR* ("Optional", "Required")=0.065. All gestures proposed for movements reached a very high consensus, while others reached a medium and a low consensus, particularly for the optional character of an annotation.

4 A Gesture Identification Study for User Interface Annotation Types

While a gesture elicitation study [73] consists of an open-ended and exploratory study to gather ideas from participants, such as about their preferences for gestures, a gesture identification study [2] consists of a closed, structured, and confirmatory study to validate ideas submitted by participants, such as their preferred gestures for some referents [20].

4.1 Method

Therefore, we conducted a gesture identification study to confirm and validate (Fig. 1-c) the previous consensus set of gestures (Fig. 9). Conducting this kind of study helps establish a common understanding among participants regarding the gestures corresponding to the annotation types being identified. By asking participants to identify which annotation they would prefer among a set of predefined *options*, it provides information on the consistency of the participants' interpretations of each annotation type. This information should improve the way we annotate certain UI elements, making them more useful in practice which impacts the decisions that need to be taken regarding the implementation of gesture recognizer.

4.1.1 Participants. Fifty-nine volunteers (n_2=55, 22 women, 37 men), aged between 18 and 57 years (M=29.10, SD=8.50, Mdn=25), from 15 different countries located in four continents, i.e., Africa, North America, Europe, and Asia) who did not participate in the early elicitation. All participants reported frequent use of computers (M=6.34, SD=1.39, Mdn=7 on a 7-point Likert scale [40]) and smartphones (M=6.03, SD=1.44, Mdn=7), less frequent use of tablets (M=2.47, SD=1.84, Mdn=2), no dexterity impairments, and had normal or corrected-to-normal vision.

4.1.2 Stimuli. For each annotation type, we have retained the three most frequently proposed gesture representations for which the agreement rate was the highest (see rightmost column of Fig. 9) to obtain a set of 18 annotations × 3 representations = 54 options. A small database of PNG files was created for this purpose.

4.1.3 Apparatus. Similarly to the GES (Section 3), we developed a QuestionPro crowd-based questionnaire that randomly presents the various representations for each annotation type and asks the participant to select the most preferred one among the set of proposals.

4.1.4 Procedure. We used a within-subject design where participants, after signing a GDPR-compliant consent form, completed the same socio-demographic questionnaire and were instructed to select the most preferred option for each annotation type among the 3 representations presented [1]. The participants were then instructed to select one option: "In the following series of questions, several options of annotation types will be displayed. Your task is to select the option that is the most appropriate according to the instruction describing which annotation type should be used."

4.2 Results and Discussion of the Identification Study

The interactive session lasted M=141.56 sec on average (SD=101.58 sec, Mdn=106 sec, Min=52 sec, Max=654 sec). Fig. 10 shows the percentage of preference for all options. Options that were never selected are omitted. For example, Q1="Which annotation would you prefer to draw attention to an element?" accepted the A51 signpost and the exclamation point in 67.80%, 32.20% of cases, respectively. Some annotation symbols were more commonly selected across the participants, indicating a shared understanding and a common ground. Based on these results, we selected 10 templates per annotation representation acquired in the elicitation study to train our classifier. We therefore transferred 10 templates × 18 annotation types = 180 samples to the classifier of Crowditivity (Fig. 1-e).

[1]Instruction: "Hello, you are invited to participate in our study related to the evaluation of annotation types. Your participation in this study is completely voluntary. There are no foreseeable risks associated with this project. It is very important for us to know your opinion on certain representations. Your responses will be strictly confidential and data from this research will be reported only in the aggregated form. Your information will be coded and will remain confidential. Thank you very much for your time and support. Please start with the study now by clicking on the Start button below."

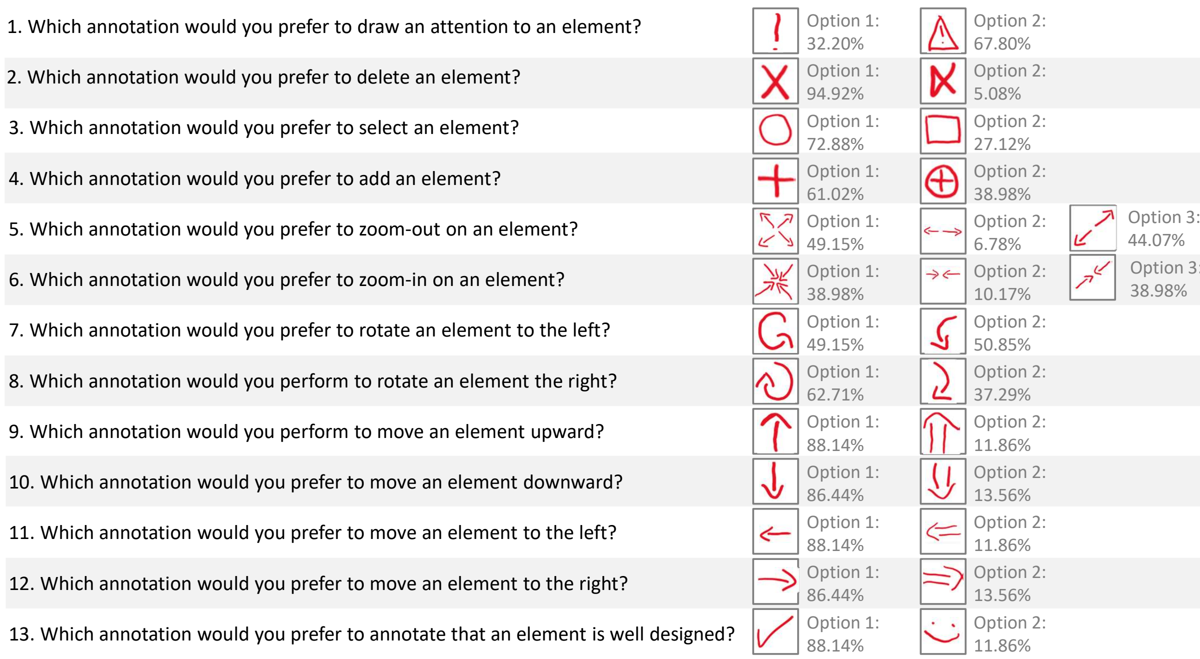


Fig. 10. Questions, options, and preferences resulting from the identification study.

5 Implementation of Crowditivity

To implement Crowditivity, we adopted a full-stack software architecture (Fig. 11) decomposed as follows. For the front-end part, we worked with:

- **ReactJS**: a popular and widely-used JavaScript library that helps in building professional UIs. ReactJS has a lot of good practices, it is easy to use, and it supports reusable components, which help in writing the code once and reuse it multiple times in different places in the application, which can significantly reduce development time. It provides good performance thanks to its implementation of Virtual DOM, whose primary objective is to enhance the overall performance of the application, unlike other traditional HTML which is built on a real DOM. It updates only the changed components, which reduces the number of renderers and improves the loading speed. ReactJS uses JSX, which is JavaScript + XML, that allows us to write HTML-like syntax in the JavaScript code. JSX can be perceived as confusing, especially in large projects. When the application grows, additional tools are required, such as extra libraries to complete the desired functionalities, which complicates.
- **Tailwind CSS**: a utility-first CSS framework that has pre-defined UI classes to be used on the fly. These pre-built classes can be used to create designs without writing all the CSS from scratch. Tailwind CSS also provides a high level of customization, allowing us to tailor the default styles of the ReactJS application. This approach means that each CSS class is specifically built to perform specific results, making it easier for us to style the UI elements. Having to think about how to create a responsive application can be time consuming and a lot of additional work in the development process, but Tailwind CSS includes responsive design classes, which allow us to create responsive UIs that can adapt to different screen sizes. However, the classes are well designed in a way that facilitates the design-for-change to the end user's needs.

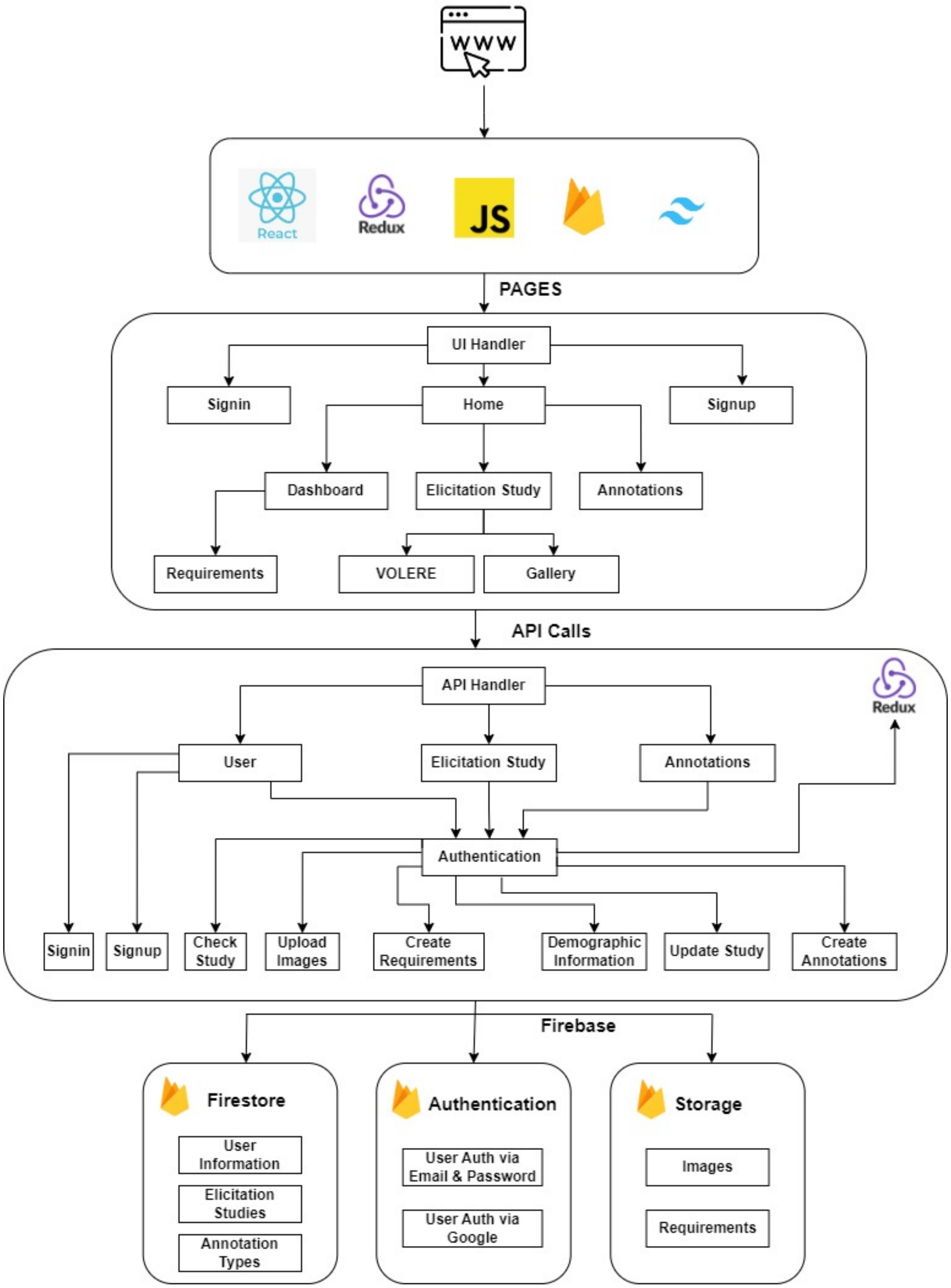


Fig. 11. Software architecture of Crowditivity.

For the back-end part of CROWDITIVITY, we decided to work with these technologies:

- **Google Firebase**: a cloud-based platform for web development that has many features such as cloud storage, authentication, and hosting. We chose Firebase because it provides a scalable infrastructure that allows us to focus on creating the website's functionalities rather than managing the database infrastructure from scratch. Firebase provides a high level of security as well as authentication features that can help protect users' data from any DDoS attacks or hackers trying to break into the application to get users' personal information. Hosting on Firebase is helpful to make the crowd-based application accessible to users when launched in production. Firebase has automatic backups and recovery features, which can help protect against data loss in case of system failure. Since the main focus is developing a crow-based application to elicit UI requirements, Firebase functions, such as "Analytics", help us to understand user behaviors, engagement, and how they are using the application to improve the user experience. A disadvantage of Firebase is that it may not be suitable for complex data queries which makes for slower query times.
- **Redux**: an easy-to-use predictable state container. It is a state management library that works along ReactJS to efficiently manage the application's state. ReactJS main concept is having stateful components that are rendered when needed, so it is like a centralized store where the state of the entire application is stored. Using Redux can be beneficial because of its ability to manage states in an organized way. When the application fetches data from Firebase, it typically makes an asynchronous API call to retrieve data, so this API call takes time. Requesting the same data multiple times may result in unnecessary API calls. Redux is a great solution for these problems by storing the fetched data in a global state container that is accessible at any time and in any component of the application. This can improve the performance of the application. A drawback of using a state management library such as Redux is that overusing it to store every state can lead to performance issues. Too much data stored can slow down the application and increase memory usage.

5.1 Gesture Recognition of Annotations

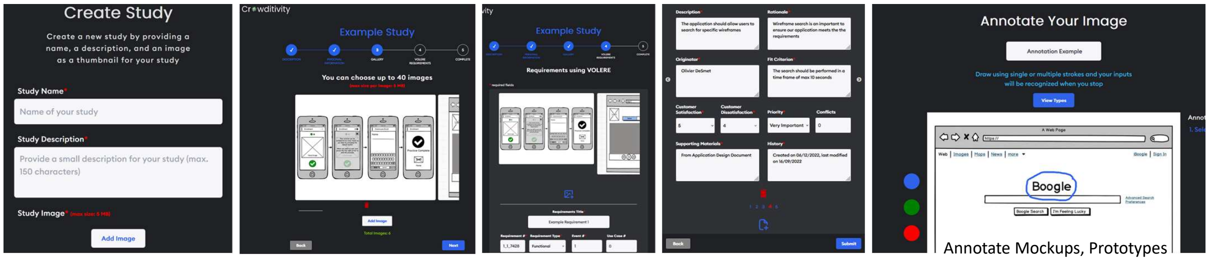
To recognize the gestures associated to the 18 annotations, we selected *$P+* [66], an updated version of the $P recognizer [68] that significantly improves recognition accuracy and execution speed for gestures produced by people, including those with low vision. *$P+* has been trained with a curated dataset of 10 templates X 18 gesture classes = 180 samples (Fig. 1-f), yielding an overall recognition rate of 98.2% with a system response time below 1 sec. This recognizer was selected for the following reasons: its accuracy is among the best recognizers [44] with a small amount of templates [66], its complexity is moderate [43], it works for both uni- and multi-stroke gestures in an articulation-independent setup. The name is derived from the *$-family* of recognizer algorithms, and the "P" means that this algorithm is a point-based recognizer so it makes use of "Point clouds". *$P+* introduced two main upgrades from its previous version: optimal alignment and exploiting the shape of the point cloud. The optimal alignment between two points works by having each point from a cloud to be matched with another point that has not been matched before because they are represented as the optimal option when points are matching. However the upgrade that was made in *$P+* was, instead of matching the first point from a cloud to a "unmatched" point from another cloud, the algorithm would now match the closest points even though they were already matched earlier, this change resulted to be faster since the time complexity has decreased by approximately **34.1%** from $O(n^{2.5})$ to $O(n^2)$. The second change was to exploit the shape structure, since $P was creating a set of 2D stroke gesture points without any ordering, this means that the algorithm disregarded any connections between point a and point b. *$P+* introduced a solution to that problem by adding angles for matching points to improve the recognition accuracy. The turning angles capture the curvature of a gesture, which is an important feature for accurate recognition.

Below we can see the pseudo-code used for the *$P+* recognizer [66]. This recognizer will recognize gestures that were drawn by comparing them to a predefined template. The normalize function is used to transform an input gesture to a standard size and position, this process makes it easier to compare with other gestures. For each template gesture, it normalizes it to the same fixed length as the input gesture and then calculates the distance between these two gestures. The minimum distance is stored in a variable "d", and if this value is less than the current score, the score is updated to "d". Finally, the output of this algorithm returns the result and the score.

```
# $P+RECOGNIZER (POINTS C, TEMPLATES templates)
n <- 32
NORMALIZE(C, n)
score <- infinity
for all T in templates do
NORMALIZE(T , n) // should be pre-processed
d <- min (CLOUD-DISTANCE(C, T ), CLOUD-DISTANCE(T,C))
if score > d then
score <- d
result <- T
return <result, score>
```

The pseudo-code shown below is related to the cloud distance algorithm which is used in *$P+* to compute the distance between two point clouds. This algorithm allows one-to-many matches, so that each point in one cloud can be matched with multiple points in another cloud. The algorithm works by having for each point in cloud C find the closest match in cloud T and stores the index of the match in an array. After that, the distance between the matched points is added to the variable called "sum". Next, for points in cloud T that were not matched, it finds the closest match in cloud C and adds the distance to the variable "sum" which is then returned as output.

```
# CLOUD-DISTANCE (POINTS C, POINTS T , int n)
matched <- new bool[n]
sum <- 0
// match points from cloud C with points from T ; one-to-many matchings allowed
for i = 1 to n do
min <- infinity
for j = 1 to n do
d <- POINT-DISTANCE(Ci , Tj )
if d < min then
min <- d
index <- j
matched[index] <- true
sum <- sum + min
// match remaining points T with points from C; one-to-many matchings allowed
for all j such that not matched[j] do
min <- infinity
for i = 1 to n do
d <- POINT-DISTANCE(Ci , Tj )
if d < min then min <- d
sum <- sum + min
return sum
```



Create and describe a UI Study Manage Mockups, Prototypes Manage and Assign VOLERE UI requirements Recognize and Classify annotations

Fig. 12. Crowditivity walkthrough for user interface requirements elicitation by crowdsourcing.

5.2 System Walkthrough and Software Architecture of Crowditivity

Crowditivity consists of a crowd-based web application where stakeholders can create, annotate, and review UI requirements during crowd elicitation (Fig. 12). Crowditivity enables any stakeholder to create a study, add a UI requirement and associated UI elements (e.g., images, screenshots, or mockups), and annotate them according to the annotation types defined. These are resulting from forward engineering (e.g., from business process analysis [60]) or reverse engineering (e.g., by recovering existing UIs [10], UI scenarios [3]) that would help them collect, manage, and understand their UIs, and review their previous work. A focus group involving three UI designers (one of them is also an experienced developer) and an HCI researcher (Fig. 1-d) elaborated and agreed on five activity diagrams (see Appendix A) expressed in Unified Modelling Language (UML) V2.5 to structure the process of Crowditivity (Fig. 12) as follows (Figs. 16 and 1-f).

5.3 Create a Study

Elicitation studies can be seen as middleware between what was designed on a mock-up and what is coded. To prevent miscommunications between the designers and the developers working on the UI, and to avoid misleading information about what will be done on this software, this stage should prevent such problems from occurring. When stakeholders want to create a study (see the activity diagram in Fig. 17), they are authenticated by email verification and are requested to add their demographic information if they want, such as first name, last name, age, gender, education, level of experience, etc. The user creates a new study by adding a title, providing a description, and adding an image that serves as a reference preview (Fig. 12). The user then follows a tutorial on how to create effective UI requirements and is enabled to add a new requirement (Fig. 18) using the Volere (Fig. 2) template extended with fields that are specifically tailored to UI requirements elicitation:

- *UI Requirement Title*: a title used to identify different requirements
- *UI Requirement ID*: a unique Identifier for each requirement
- *UI Requirement Type*: functional or non-functional requirement
- *Event number*: list of events
- *Use case number*: use cases that need this specific requirement if any
- *Description*: a one-sentence statement of the intention of the requirement
- *Rationale*: a justification of the requirement
- *Originator*: the stakeholder who raised this requirement
- *Fit criterion*: a measurement to test if the solution matches the original requirement
- *Customer satisfaction*: the degree of stakeholder's happiness (from 1 to 5) if this requirement is implemented
- *Customer dissatisfaction*: the degree of stakeholder's unhappiness (from 1 to 5) if this requirement is not implemented

- *Priority*: the relative importance of the requirement (Not important, Important, Very important, and Crucial)
- *Conflicts*: if another requirement affects this requirement
- *Supporting Materials*: pointer to document that illustrates and explains this requirement
- *History*: creation date and if any changes were made to this requirement

At any time, the user can upload any image of a mockup, such as a wireframe or a sketch, which automatically generates a UI requirement template by default. If needed, the main image can be enriched with 5 additional images to depict a scenario, a task (e.g., according to a task model), or a sequence of actions. New UI requirements can also be attached to these images at any time. In sum, an image containing UI elements can be assigned to one to many UI requirements and a UI requirement can be assigned to many images (Fig. 39).

5.4 Add an Annotation

As soon as a study is ready, Crowditivity generates a URL to be shared to annotators who can connect and start annotating the mock-ups and their UI elements by gestures corresponding to the 18 annotation types (Fig. 9). Since color coding is largely utilized in studies for feedback [4], the annotator can annotate using three different colors, i.e., red, green, and blue, based on how important each change is: "red" indicates that this change has the highest priority among all annotations, "green" and "blue" indicate that this change has a normal and low priority, respectively. The ability to annotate is important in UI requirements engineering because it allows designers to highlight specific features relevant to the UI itself [22,25]. Adding an annotation on a UI element is characterized by (Fig. 39):

- *Input mechanism*: the name of the devices used for gesture input (e.g., a mouse, a trackpad)
- *Document type*: the format of the document being annotated (e.g.,.png, .jpeg, .jpg, .svg)
- *Overview of adding annotations*: a collection of multiple user-generated gesture inputs on an image to be recognized
- *Annotation types recognized*: a list of recognized gestures with the name of the annotation type

Crowditivity stores an annotation with 3 attributes:

- *Annotation name*: the user's name for the image that will be annotated
- *Marking up*: the marking of the annotation based on different colors (i.e., red, green, or blue)
- *Data transfer*: the link to all annotations produced by an annotator, which will be accessible on the user's "My Annotations" page.

5.5 Review the Study

We discussed the functionalities of Crowditivity, a crowd-based web application that enables users to design their studies and annotate various types of images representing the mockups and their UI elements, thus facilitating a deeper understanding of the UI, which may require modifications in its design. Within Crowditivity, each user is assigned a personal account to create studies and annotations, allowing them to track their activities effectively. This feature provides users with visibility into their progress, fostering improved engagement. As a result, users are likely to feel more invested in their experience, which enhances their motivation to continue using the application. Crowditivity enables its users to access, edit, and check the user studies they created, the annotations they produced, the UI requirements, and the respective images that were created in each study.

6 Evaluation Experiment

After outlining the software's design (Fig. 1-d) and describing its implementation (Fig. 1-f) including its classifier (Fig. 1-e), we are now in a position to assess how designers perceive its use (Fig. 1-g). To evaluate Crowditivity, we conducted an experiment with 8 participants who did not participate in the previous studies (1 woman, 7 men, aged from 19 years to 37 years), who already contributed to some UI design process, with moderate to extensive experience (Fig. 1-g). For this purpose, we created a case study with Crowditivity (see video as supplementary material) in which participants were instructed to enter 2 requirements, to associate them to 3 images resulting from a wireframe process with Balsamiq, and to annotate them freely. At the end of this process, participants were asked to fill out a UEQ+ form, a questionnaire resulting from applying the modular evaluation method UEQ [57]. UEQ+ is a modular extension of the User Experience Questionnaire (UEQ) [56], which is an end-user questionnaire to measure user experience quickly, simply, and immediately while covering a comprehensive impression of the user experience [57]. As a modular extension, UEQ+ has more user experience (UX) scales compared to UEQ and can be used to evaluate products in special scenarios. UEQ+ is designed to gather feedback from users about their subjective experience. In addition, the effort to set up a UEQ+ for a project can create a deeper common understanding of the UX requirements and their importance in the project team and create additional value. We chose to measure the user experience using 9 scales (each scale consists of four statements to evaluate on a 7-point Likert: 1=worst, 7=best):

- Attractiveness: Overall impression of Crowditivity. Do users like or dislike it?
- Efficiency: Can users solve their tasks without unnecessary effort? Does it react fast?
- Stimulation: Is it exciting and motivating to use Crowditivity? Is it fun to use?
- Novelty: Is the UI design creative? Does it catch the interest of users?
- Usefulness: Does using Crowditivity bring advantages?
- Value: Does Crowditivity design look professional and of high quality?
- Visual Aesthetics: Does Crowditivity look beautiful and appealing?
- Intuitive Use: Can Crowditivity be used immediately without any training or help?
- Clarity: Impression towards order, structure and visual complexity of a graphical user interface

6.1 Quantitative and Qualitative Measures

We evaluate a series of scales, where: $\forall$ scale $S_i, \forall i \in \{1, \ldots, n\}$ decomposed into m_i sub-scales $SC_{ij}, \forall j \in \{1, \ldots, m_i\}$ or items to be evaluated (e.g., the scale Attractiveness is decomposed into four sub-scales: annoying *vs.* enjoyable, bad *vs.* good, unpleasant *vs.* pleasant, and unfriendly *vs.* friendly), each sub-scale SC_{ij} being a differential scale with 7 points between items of each pair (e.g., annoying o o o o o o o enjoyable). We measure each sub-scale SC_{ij} employing a 7-point Likert scale with response categories "Strongly disagree" (=1) to "Strongly agree" (=7). For each S_i, we define:

(1) Subscale-Score (SS): an ordinal integer variable that measures the score for each sub-scale SC_{ij} of a scale S_i, ranging from 1="Strongly disagree" to 7="Strongly agree".
(2) Subscale-Importance (SI): an ordinal integer variable that measures the weight of each sub-scale SC_{ij} of a scale S_i, ranging from 1="Least important" to 7="Most important".
(3) Scale-Mean-Score (SMS): a real variable that measures the average score obtained on all sub-scales SC_{ij} of a scale SC_i for all participants, ranging from -3="Mostly negative" to +3="Mostly positive" calculated as follows: $\mathrm{SMS}(SC_i) = \frac{\sum_{j=1}^{m} \mathrm{SS}(SC_{ij})}{n} - 4, \forall i \in \{1, \ldots, n\}$, where 4 is the scale median.
(4) Scale-Mean-Importance (SMI): a real variable that measures the average weight of importance of all sub-scales SC_{ij} of a scale SC_i for all participants, ranging from -3="Mostly negative" to +3="Mostly positive" computed as follows: $\mathrm{SMI}(SC_i) = \frac{\sum_{j=1}^{m} \mathrm{SS}(SI_{ij})}{n} - 4, \forall i \in \{1, \ldots, n\}$, where 4 is the scale median.

6.2 Tasks

We devised four tasks to be carried out by each participant:

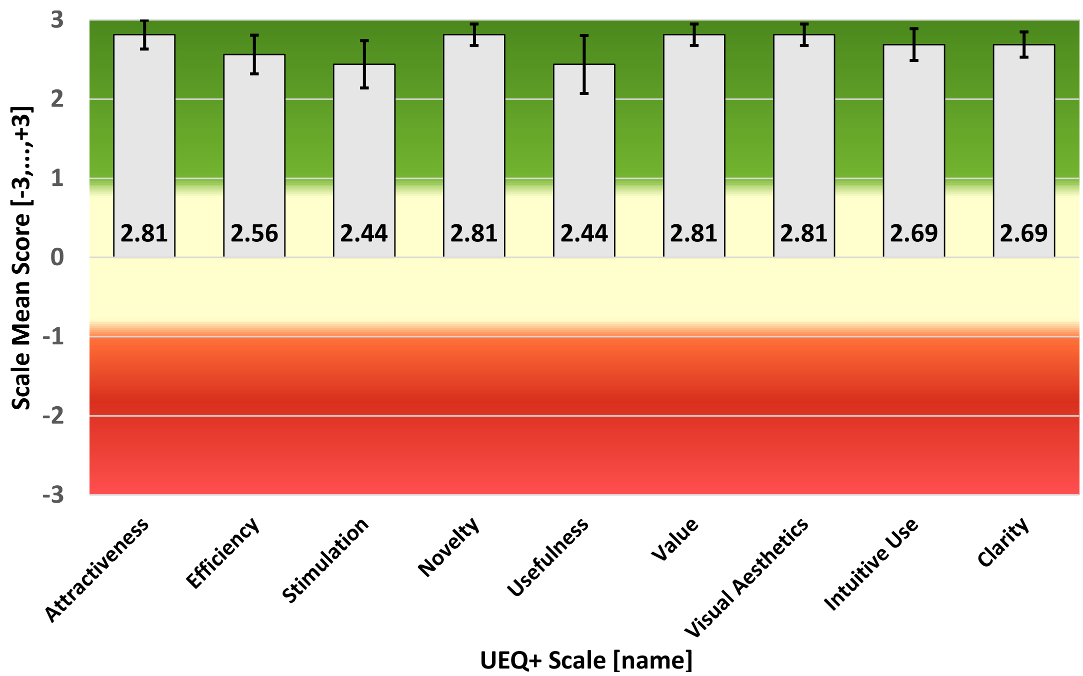


Fig. 13. Scale-Mean-Scores of the UEQ+ evaluation of Crowditivity. Error bars show a confidence interval of 95%.

(1) A *discovery task*, in which participants received a tutorial on using Crowditivity and then interacted freely for about five minutes.
(2) A *practicing task*, in which participants were instructed to practice gestures on sample mock-ups to learn the 18 annotation types. We have developed a practice module enabling each participant to practice the gestures corresponding to the 18 types of annotation to learn them and observe the correct operation of the recognizer (see Appendix B for screenshots for all annotation types).
(3) An *annotation task*, in which participants discovered a case study for a web application and were instructed to annotate the mock-ups corresponding to the UI requirements. No particular order was imposed. We suggested, but not imposed, a time limit of 10 minutes. We recorded the session time in sec. Crowditivity captured raw data about annotations and their recognition for classification
(4) An *evaluation task*, in which participants were instructed to fill in a UEQ+ questionnaire with the 9 scales covering general aspects, pragmatic, and hedonic qualities [57]. Free comments could be added. UEQ+ explicitly covers pragmatic and hedonic dimensions of user experience, not just usability, it consists of several scales to be analyzed individually and globally, there are analysis instruments and published norms for interpreting their results, which is not the case for other similar methods.

Fig. 13 and Fig. 14 show this evaluation's scale mean scores and importance, respectively. The feedback from the questionnaire showed that all UEQ+ scales have shown positive results, with mean values close to 3 on a scale of -3 to +3. This indicates that users have given favorable responses across the different aspects of the user experience.

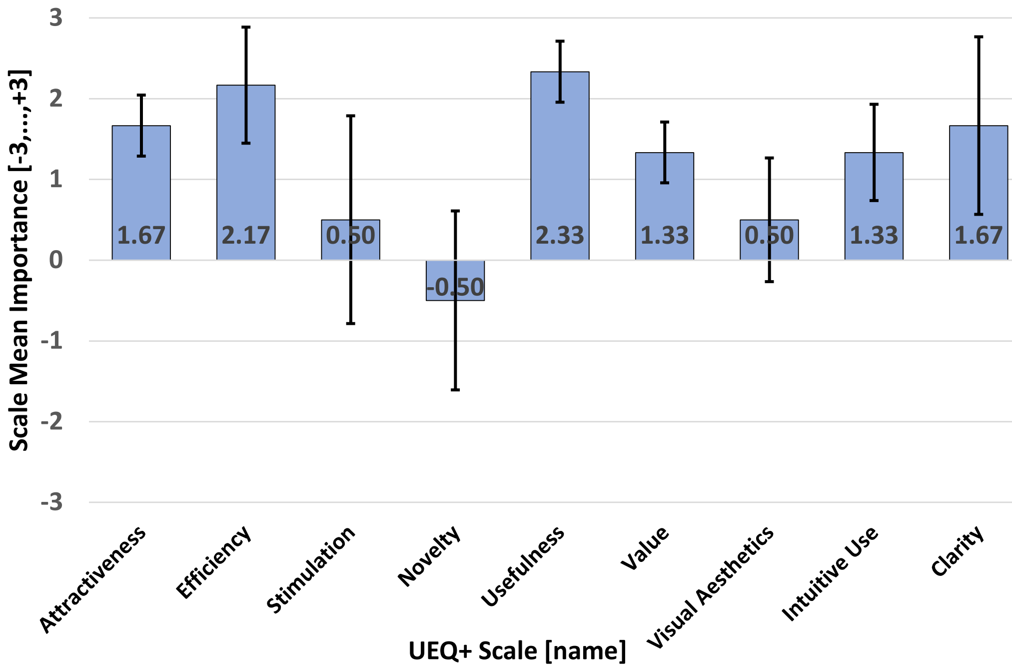


Fig. 14. Scale-Mean-Importance of the UEQ+ evaluation of Crowditivity. Error bars show a confidence interval of 95%.

6.3 Results and Discussion

The scales that received the highest scale mean scores are Value (*M*=2.81, *SD*=0.39), Novelty (*M*=2.81, *SD*=0.39), Visual Aesthetics (*M*=2.81, *SD*=0.39), and Attractiveness (*M*=2.81, *SD*=0.53). These scores indicate that participants got a positive outcome, enjoyed its visual design, and perceived a real value from the tasks they performed with Crowditivity. While these two last scales were evaluated very positively, they were not ranked as very important: Novelty (*M*=-0.50, *SD*=1.38) and Visual Aesthetics (*M*=0.50, *SD*=0.96) received the two lowest scale mean importance. Some participants reported that they were very familiar with manual annotations (hence, the low importance), so they appreciated the novel support of Crowditivity in that sense (hence, the high mean score). The narrow standard deviation of these scores further suggests a strong consensus among participants. Indeed, we computed Cronbach's α=0.96, Spearman' ρ (Halves=0.88, Odd-Even=0.99), and Guttman's reliability coefficient λ (Halves=0.88, Odd-Even=0.98), which all demonstrate an excellent consensus and reliability of the answers, which is partially explained by their small number. Intuitive Use (*M*=2.69, *SD*=0.58) and Clarity (*M*=2.69, *SD*=0.46) suggest that participants found that Crowditivity is well structured, easy to use, and balanced the user's needs with innovative design.

Finally, Efficiency (*M*=2.56, *SD*=0.70), Stimulation (*M*=2.44, *SD*=0.86), and Usefulness (*M*=2.44, *SD*=1.06) have slightly lower mean scores, which shows a potential room for some improvements regarding the user experience in these areas. Nonetheless, these scores are considered quite high and some enhancements can be made to the usefulness of tasks performed. Some participants reported that Efficiency was slower than expected because of the gesture recognition required for annotation classification. Although they sometimes repeated the gesture for positive recognition, they admitted the value of this process as they forgot that the recognition is

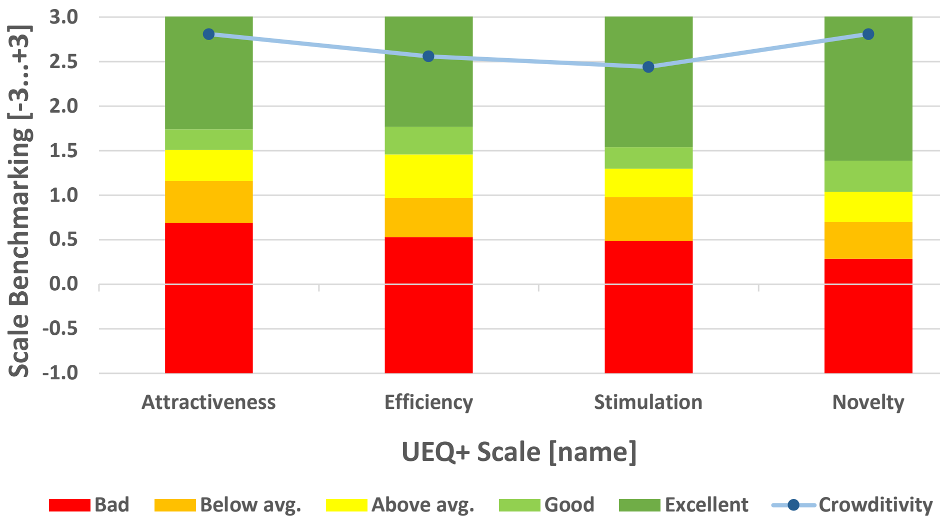


Fig. 15. Benchmarking of scale mean scores of some scales of Crowditivity.

optional. This explains why Usefulness (M=2.33, SD=0.47) and Efficiency (M=2.17, SD=0.90) are the two most important scales, despite their mean scores.

Clarity (M=1.67, SD=0.37), Attractiveness (M=1.67, SD=0.47), Value (M=1.33, SD=0.47), and Intuitive Use (M=1.33, SD=0.75) received an average value for their importance, which are typical values. Stimulation (M=0.50, SD=1.61) was assessed as less important than the previous scales as they recognized that annotating per se is not always the most stimulating activity. They tend to prefer reporting feedback orally in person instead of graphically remotely. The scores received for importance received a reasonable value of consistency and reliability, but less than the scores themselves: Guttman's reliability $\lambda = 0.89$ (Halves=0.58, Odd-Even=0.51). Schrepp et al. [56] mention more precise intervals to interpret some scales based on benchmarking performed on a set of evaluations. This benchmarking decomposes each scale into five intervals: bad, below average, above average, good, and excellent. Fig. 15 shows the interval in which our five concerned scales, i.e., Attractiveness, Efficiency, Stimulation, and Novelty are falling, which all belong to the "Excellent" category.

7 Statistics Describing the Behavior of Annotators

Computing statistics that describe the behavior of annotators allows designers and developers to uncover problem patterns (e.g., problems that are recurring), identify usability issues (e.g., incorrectly designed UI elements), and optimize the design to meet user needs (e.g., sub-optimal layout that does not correspond to the task flow). By analyzing factors such as annotation accuracy, time spent on tasks, and error rates, designers can pinpoint areas for improvement. For this purpose, we define a set of 10 metrics describing the behavior of annotators, as an output of Crowditivity:

(1) Number of annotators (N): total number of participants who annotated at least one UI element of the study (e.g., N=8 in our evaluation).
(2) Number of annotations (A): total number of annotations per UI element of a mockup, per mockup or per prototype of a study, per study, or across studies (e.g., A=16 annotations per mockup).
(3) Frequency of annotator (FN): ratio of the number of participations of an annotator per unit of time or per time interval. Based on this metric, different categories can be defined such as regular user ($FN \geq 50\%$

for a participant who contributes to half of the studies s/he invited to), occasional user, few-time user, and one-shot user (a user that contributes to only one study).
(4) Frequency of annotations (*FA*): ratio of the number of annotations per unit of time or per time interval.
(5) Distribution of annotations per annotator: average number of annotations produced by an annotator per UI element, mockup, or study. The distribution is completed with the average number, the standard deviation, and the median (e.g., a one-shot user produced 3.6 annotations per mockup of a study (M=3.6, SD=4.4, Mdn=2).
(6) Distribution of annotations over time: average number of annotations produced for a study over the requested period of time, with standard deviation, and the median (e.g., a one-month study received $M = 10.6$ annotations on average with a SD=7.7 and a median of Mdn=8).
(7) Distribution of annotators over time: average number of annotators produced for a study over the requested period of time, with standard deviation, and the median.
(8) Annotation duration (*AD*): number of days elapsed between the day of the first annotation and the day of the last annotation that has been made for a study (e.g., AD=7 days).
(9) Annotation recency (*AR*): number of days elapsed since the last annotation has been produced for a study (e.g., AR=11 days to indicate that the last annotation of a study was made already 11 days ago).
(10) Number of strokes per annotation (*NS*): total number of gesture strokes (uni-stroke and multi-stroke) produced per annotation (e.g., NS=5 strokes per annotation).

This metric-based approach not only enhances the overall user experience but also leads to more effective collaboration between annotators and practitioners.

8 Limitations of Crowditivity and Threats to Validity

8.1 Limitations

In the past years, several application domains were explored using the Crowditivity platform to investigate gestural annotation of user interface requirements in crowd elicitation:

- CareFlow: an innovative, user-friendly platform for managing medical appointments between patients and physicists (7 UI requirements for the patient, 9 requirements for the doctor, 16 user stories, 20 associated mock-ups for the smartphone, 18 associated mock-ups for the desktop version, N=4 annotators, A=3.2 annotations per mock-up on average over one week, FN=100% because two annotators were assigned to each version, i.e., 2 for the smartphone and 2 for the desktop, NS=4.5 strokes per annotation on average with an average of 14.4 strokes per mock-up).
- SmartGarden: an application providing anyone with the essential knowledge to create and maintain their vegetable garden with ease.
- Make it!: an application simplifying users' experience in completing DIY projects with solutions tailored to their needs and skill levels.
- Buddy's Budd: an application offering a configurable matching mechanism that allows users to connect with each other based on availability and common interests).
- FeedGuide: an application that will enable employees to anonymously express their feelings without fearing any consequences and therefore improve the company's atmosphere and performance.
- QuickSpot: an application to centralize all parking related information, such as availability and planning, in a single, user-friendly application.

Collectively, these projects demonstrate the applicability of Crowditivity across diverse domains and confirm its adoption potential in practical design contexts. Despite this diversity, several limitations remain. Crowditivity primarily focuses on the recognition and classification of annotation types by stroke gestures rather than the automatic identification of UI elements within mockups, such as in UIskei [58] and SketchiXML [16].

Consequently, while the system classifies annotations associated with UI components, it does not independently detect or classify the components themselves. Complementary approaches exist to address this gap. Semi-automatic workflows, such as the "Find–Draw–Verify" pattern, support crowdsourced detection of UI elements by identifying their presence, delineating them with bounding boxes, and validating their type. Fully automated solutions based on computer vision techniques enable online or offline identification and segmentation of UI elements directly from interface images. In-HITT[28] performs crowdsourcing of annotations of UI elements from mockups according to the "Find-Draw-Verify" workflow pattern: find the presence of UI element in the image, draw a bounding box around it, and verify its type) or completely automatically (e.g., VisionAPI [12] performs on-line or off-line identification and segmentation of UI elements. Integrating gestural annotation recognition with automated UI element detection would enable a more comprehensive classification pipeline, producing structured outputs readily exploitable by designers and developers.

User studies further revealed behavioral considerations regarding gesture recognition. Although gesture-based annotation was encouraged, it remained optional. Some participants reported forgetting that automatic gesture recognition was available, while others deliberately preferred optional recognition to preserve the spontaneity and natural flow of annotation. These findings suggest that system transparency and interaction design play a critical role in balancing automation with user autonomy.

Currently, Crowditivity supports two annotation modalities: textual input and graphical annotations produced through gestures. Extending the platform to accommodate additional media types, such as audio recordings, video annotations, or web links [22], could broaden expressive capabilities and increase accessibility. Preliminary observations indicate that voice annotations, in particular, may offer promising opportunities for richer requirement capture.

From a technical perspective, Crowditivity automatically recognizes and classifies 18 gesture classes with multiple articulatory variations, demonstrating flexibility and robustness in stroke interpretation. However, the system does not currently support established stroke-based symbol systems commonly used in document editing and proofreading, including shorthand and standardized editorial notation. But Crowditivity does not support other stroke gestures or signs that are typically used for document editing and proofreading, such as those defined in shorthand, stenography, and Oxford editing and proofreading symbols. Incorporating such symbol catalogs would require retraining recognition models on sufficiently large and representative stroke template datasets. Expanding gesture coverage in this manner could enhance expressiveness and align the system more closely with established annotation practices.

8.2 Threats to External Validity

For the qualitative evaluation reported in Section 6, we adopted a convenience sampling which produces diverse samples, but in a limited way. So this part of our evaluation cannot claim generalizability. Instead, we aimed to provide insights into how the involved participants would assess the annotation process as supported by Crowditivity. Future experiments should confirm these insights with more diverse populations of practitioners. For this purpose, we could conduct a similar experiment with a stratified or cluster sampling based on the nine development expertise levels of Costabile et al. [15]. Secondly, our evaluation is based on a single web application chosen for its simplicity, potentially leading to disparities between the annotations obtained for this case study and other real-world UIs. For the same reasons, we cannot generalize to other applications. Thirdly, our user study was conducted remotely to satisfy the crowd-based criterion [1], which does not ensure control and consistency of the experimental conditions. Nothing prevents us from reproducing the experiment in a more controlled setup.

8.3 Threats to Internal Validity

Our participants may have found the scales and items somewhat difficult to grasp and differentiate. We tried to mitigate this aspect by explaining their difference independently of any particular case study. To ensure the natural character of the experiment, we did not impose any time constraints on participants, which may not reflect the real usage conditions.

8.4 Threats to Construct Validity

A within-subject design was always used for our experiments, which could affect the validity of the results. For example, the evaluation could be redefined as a between-subject design if participants are annotating UIs of different projects. Our evaluation focused on the Crowditivity UX, not on the potential benefits brought by its results to stakeholders. For example, we did not produce any table summarizing the metrics that would be ready for designers to re-consider and for developers to re-develop. This experiment is beyond the scope of this paper and could represent a real asset to identify the positive impact of the annotation process as supported by Crowditivity.

9 Conclusion and Future Work

In this paper, we present Crowditivity, a freely accessible online web application that supports the process of user interface requirements elicitation by enabling stakeholders emerging from the crowd (the number of stakeholders per study is unlimited) to edit studies, UI requirements in general and in Volere formats, UI elements and to link them. In this way, any stakeholder can annotate the fragment. Contrarily to similar software, the annotations are automatically recognized based on a 2D multi-stroke gesture recognizer which has been trained on gesture templates. These templates were captured during a gesture elicitation study followed by an identification study that determined the most preferred representations as gestures for each of the 18 annotation types. While the overall evaluation based on the UEQ+ method returned encouraging results, some progress needs to be made regarding the definition of UI requirements. New artifacts such as empathy maps [31] and personas [32] are now being used during requirements elicitation. These artifacts should be incorporated in Crowditivity in the future. UI requirements should not be limited to graphical UIs, but could be extended to other interaction modalities, such as vocal UIs or haptic UIs [36], which probably requires new attributes of the template. Nowadays, the development life cycle is more agile, thereby involving a collaborative prototyping tool to create the prototype with a strong mapping with user stories, then crowdsourcing to improve the UI of the prototype by annotation and by modification.

Acknowledgments

The authors thank the participants of the gesture studies reported in the paper for their participation. Jean Vanderdonckt is supported by the EU EIC Pathfinder-Awareness Inside challenge "Symbiotik" project (1 Oct. 2022-30 Sep. 2026) under Grant no. 101071147. We also acknowledge the support of the "User Interface Adaptation through User-Experience-based Reinforcement Learning" (AKILA) project (1 January 2022-31 December 2024) under Grant no. CIAICO/2021/303 between Universitat Politècnica de València (Spain) and Université catholique de Louvain (Belgium) as well as the "User-Centered Intelligent Software Systems Adaptation" (UCI-Adapt) project granted by the Spanish State Research Agency, Ministry of Science and Innovation (1 September 2023-31 August 2026) under Grant no. PID2022-140106NB-I00 between the Universitat Politècnica de València (Spain), Universidade Nova de Lisboa (Portugal), and the Université catholique de Louvain (Belgium).

References

[1] Abdullah X. Ali, Meredith Ringel Morris, and Jacob O. Wobbrock. 2018. Crowdsourcing Similarity Judgments for Agreement Analysis in End-User Elicitation Studies. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology* (Berlin, Germany) *(UIST '18)*. Association for Computing Machinery, New York, NY, USA, 177–188. https://doi.org/10.1145/3242587.3242621

[2] Abdullah X. Ali, Meredith Ringel Morris, and Jacob O. Wobbrock. 2019. Crowdlicit: A System for Conducting Distributed End-User Elicitation and Identification Studies. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (Glasgow, Scotland, UK) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300485

[3] Asmaa Alsumait. 2004. *User interface requirements engineering : a scenario-based framework.* Ph. D. Dissertation. Gina Cody School of Engineering and Computer Science, Concordia University, Montréal, Québec, Canada. https://spectrum.library.concordia.ca/id/eprint/8009/

[4] Christine Alvarado, Andy Kearney, Alexa Keizur, Calvin Loncaric, Miranda Parker, Jessica Peck, Kiley Sobel, and Fiona Tay. 2015. *LogiSketch: A Free-Sketch Digital Circuit Design and Simulation SystemLogiSketch.* Springer International Publishing, Cham, 83–90. https://doi.org/10.1007/978-3-319-15594-4_8

[5] Caroline Appert and Shumin Zhai. 2009. Using Strokes as Command Shortcuts: Cognitive Benefits and Toolkit Support. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (Boston, MA, USA) *(CHI '09)*. Association for Computing Machinery, New York, NY, USA, 2289–2298. https://doi.org/10.1145/1518701.1519052

[6] Nathalie Aquino, Jean Vanderdonckt, Nelly Condori-Fernández, Óscar Dieste, and Óscar Pastor. 2010. Usability Evaluation of Multi-Device/Platform User Interfaces Generated by Model-Driven Engineering. In *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement* (Bolzano-Bozen, Italy) *(ESEM '10)*. Association for Computing Machinery, New York, NY, USA, Article 30, 10 pages. https://doi.org/10.1145/1852786.1852826

[7] Brian P. Bailey and Joseph A. Konstan. 2003. Are informal tools better? comparing DEMAIS, pencil and paper, and authorware for early multimedia design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, USA) *(CHI '03)*. Association for Computing Machinery, New York, NY, USA, 313–320. https://doi.org/10.1145/642611.642666

[8] Hani Bani-Salameh and Nadera Al jawabreh. 2015. Towards a Comprehensive Survey of the Requirements Elicitation Process Improvements. In *Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication* (Batna, Algeria) *(IPAC '15)*. Association for Computing Machinery, New York, NY, USA, Article 60, 6 pages. https://doi.org/10.1145/2816839.2816872

[9] Daniel Bildhauer, Tassilo Horn, and Jurgen Ebert. 2009. Similarity-Driven Software Reuse. In *Proceedings of the ICSE Workshop on Comparison and Versioning of Software Models (CVSM '09)*. IEEE Computer Society Press, Los Alamitos, USA, 31–36. https://doi.org/10.1109/CVSM.2009.5071719

[10] Laurent Bouillon, Quentin Limbourg, Jean Vanderdonckt, and Benjamin Michotte. 2005. Reverse engineering of Web pages based on derivations and transformations. In *Proceedings of the Third Latin American Web Congress* (Buenos Aires, Argentina) *(LA-WEB'2005)*. IEEE Computer Society Press, Los Alamitos, USA, 1–11. https://doi.org/10.1109/LAWEB.2005.29

[11] Kyle Boyd, Patrick McAllister, Maurice Mulvenna, Raymond Bond, Hui Wang, Ivor Spence, Guanfeng Wu, and Abbas Haider. 2023. Designing Multimodal Video Search by Examples (MVSE) user interfaces: UX requirements elicitation and insights from semi-structured interviews. In *Proceedings of the European Conference on Cognitive Ergonomics 2023* (<conf-loc>, <city>Swansea</city>, <country>United Kingdom</country>, </conf-loc>) *(ECCE '23)*. Association for Computing Machinery, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/3605655.3605665

[12] Paul Brie, Nicolas Burny, and Jean Vanderdonckt. 2023. VisionAPI: An API for Offline and Online Segmentation and Identification of Hand-Sketched Graphical User Interfaces. In *Companion Proceedings of the 2023 ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Swansea, United Kingdom) *(EICS '23 Companion)*. Association for Computing Machinery, New York, NY, USA, 59–67. https://doi.org/10.1145/3596454.3597184

[13] Georg Brun. 2020. Conceptual re-engineering: from explication to reflective equilibrium. *Synth.* 197, 3 (2020), 925–954. https://doi.org/10.1007/S11229-017-1596-4

[14] Yang Chen, Jing Yang, Scott Barlowe, and Dong H. Jeong. 2010. Touch2Annotate: generating better annotations with less human effort on multi-touch interfaces. In *Proceedings of ACM Int. Conf. on Human Factors in Computing Systems - Extended Abstracts* (Atlanta, Georgia, USA) *(CHI EA '10)*. Association for Computing Machinery, New York, NY, USA, 3703–3708. https://doi.org/10.1145/1753846.1754042

[15] Maria Francesca Costabile, Piero Mussio, Loredana Parasiliti Provenza, and Antonio Piccinno. 2008. End Users As Unwitting Software Developers. In *Proceedings of the 4th International Workshop on End-user Software Engineering* (Leipzig, Germany) *(WEUSE '08)*. Association for Computing Machinery, New York, NY, USA, 6–10. https://doi.org/10.1145/1370847.1370849

[16] Adrien Coyette, Sascha Schimke, Jean Vanderdonckt, and Claus Vielhauer. 2007. Trainable Sketch Recognizer for Graphical User Interface Design. In *Human-Computer Interaction - INTERACT 2007, 11th IFIP TC 13 International Conference, Rio de Janeiro, Brazil, September 10-14, 2007, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 4662)*, Maria Cecília Calani Baranauskas, Philippe A. Palanque, Julio Abascal, and Simone Diniz Junqueira Barbosa (Eds.). Springer, 124–135. https://doi.org/10.1007/978-3-540-74796-3_14

[17] Yngve Dahl, Kshitij Sharma, and Dag Svanæs. 2025. Facilitation Skills in Participatory Design. In *Proceedings of the ACM Int. Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 802, 13 pages. https://doi.org/10.1145/3706598.3713658

[18] Xavier de Ryckel, Arthur Sluÿters, and Jean Vanderdonckt. 2022. SnappView, a Software Development Kit for Supporting End-user Mobile Interface Review. *Proc. ACM Hum.-Comput. Interact.* 6, EICS, Article 173 (jun 2022), 38 pages. https://doi.org/10.1145/3534527

[19] Robert A. Elliott and Edward B. Allen. 2013. A methodology for creating an IEEE standard 830-1998 software requirements specification document. *J. Comput. Sci. Coll.* 29, 2 (Dec. 2013), 123–131.

[20] Adina Friedman and Francesco Cafaro. 2024. Designing Gestures for Data Exploration with Public Displays via Identification Studies. *Information* 15, 6 (2024), 22 pages. https://doi.org/10.3390/info15060292

[21] Bogdan-Florin Gheran, Jean Vanderdonckt, and Radu-Daniel Vatavu. 2018. Gestures for Smart Rings: Empirical Results, Insights, and Design Implications. In *Proceedings of the ACM Int. Conf. on Designing Interactive Systems (DIS '18)*. Association for Computing Machinery, New York, NY, USA, 623–635. https://doi.org/10.1145/3196709.3196741

[22] Jonathan Grudin and David Bargeron. 2005. *Multimedia Annotation: An Unsuccessful Tool Becomes a Successful Framework* (communication and collaboration support systems ed.). Technical Report. 62–76 pages. https://www.microsoft.com/en-us/research/publication/multimedia-annotation-unsuccessful-tool-becomes-successful-framework/

[23] Mieke Haesen, Davy Vanacken, Kris Luyten, and Karin Coninx. 2016. Storyboards as a Lingua Franca in Multidisciplinary Design Teams. In *Collaboration in Creative Design, Methods and Tools*, Panos Markopoulos, Jean-Bernard Martens, Julian Malins, Karin Coninx, and Aggelos Liapis (Eds.). Springer, Berlin, 211–231. https://doi.org/10.1007/978-3-319-29155-0_10

[24] Jean-Luc Hak, Olivier Nicolas, Philippe A. Palanque, and Marco Winckler. 2019. An Attempt to Fathom the Role of Annotations in User-Centered Design Process. In *Joint Proceedings HCI Engineering 2019 - Methods and Tools for Advanced Interactive Systems and Integration of Multiple Stakeholder Viewpoints co-located with 11th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2019), Valencia, Spain, June 18, 2019 (CEUR Workshop Proceedings, Vol. 2503)*, Benjamin Weyers and Judy Bowen (Eds.). CEUR-WS.org, 139–146. https://ceur-ws.org/Vol-2503/paper2_7.pdf

[25] Jean-Luc Hak, Marco Winckler, and David Navarre. 2016. PANDA: prototyping using annotation and decision analysis. In *Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Brussels, Belgium) *(EICS 2016)*, Kris Luyten and Philippe A. Palanque (Eds.). Association for Computing Machinery, New York, NY, USA, 171–176. https://doi.org/10.1145/2933242.2935873

[26] Thomas Hathaway and Angela Hathaway. 2016. *Requirements Elicitation Techniques - Simply Put!: Helping Stakeholders Discover and Define Requirements for IT Projects.* CreateSpace Independent Publishing Platform, South Carolina, United States.

[27] Marc Hesenius, Mak Krvavac, Valbjörn Jón Valbjörnsson, Theresia Mita Erika, and Matthias Book. 2025. How To Draw Commands? An Elicitation Study for Sketching on Spreadsheets. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 848, 31 pages. https://doi.org/10.1145/3706598.3715269

[28] Yi-Ching Huang, Chun-I Wang, Shih-Yuan Yu, and Yung-jen Hsu. 2013. In-HIT Example-Guided Annotation Aid for Crowdsourcing UI Components. In *Proceedings of the 1st AAAI Conference on Human Computation and Crowdsourcing* (Palm Springs, California, USA) *(HCOMP '13, Vol. 1)*, Björn Hartmann and Eric Horvitz (Eds.). Association for the Advancement of Artificial Intelligence (AAAI) Press, Washington, DC, USA, 91–92. https://doi.org/10.1609/hcomp.v1i1.13052

[29] Md Athar Imtiaz, Rachel Blagojevic, Andrew Luxton-Reilly, and Beryl Plimmer. 2017. A survey of intelligent digital ink tools use in STEM education. In *Proceedings of the Australasian Computer Science Week Multiconference* (Geelong, Australia) *(ACSW '17)*. Association for Computing Machinery, New York, NY, USA, Article 13, 8 pages. https://doi.org/10.1145/3014812.3014825

[30] ISO. 2018. *ISO/IEC/IEEE 29148:2018 - Systems and software engineering Life cycle processes Requirements engineering.* standard. International Standard Organization, Geneva. https://www.iso.org/standard/72089.html

[31] Ezequiel Kahan, Emilio Insfrán, Marcela Genero, and Alejandro Oliveros. 2021. Studying the Influence of Empathy Maps on Brainstorming for Requirements Elicitation: A Quasi-Experiment. In *Proceedings of 29th Int. Conf. on Information Systems Development: Crossing Boundaries between Development and Operations (DevOps) in Information Systems* (Valencia, Spain) *(ISD 2021)*, Emilio Insfrán, Silvia Abrahão, Marta Fernández, Chris Barry, Michael Lang, Henry Linger, and Christoph Schneider (Eds.). Universitat Politècnica de València / Association for Information Systems, Atlanta, GA, USA. https://aisel.aisnet.org/isd2014/proceedings2021/methodologies/16

[32] Ezequiel Kahan, Emilio Insfrán Pelozo, Marcela Genero, and Alejandro Oliveros. 2023. Improving the Requirement Elicitation Process using Empathy Maps and Personas: A Quasi-Experiment. In *Anais do WER23 - Workshop em Engenharia de Requisitos, Porto Alegre, RS, Brasil, Agosto 15-17, 2022*, Leandro Antonelli, Márcia Lucena, and Roxana L. Q. Portugal (Eds.). LFS (UFRN, Brasil). https://doi.org/10.29327/1298356.26-2

[33] Massila Kamalrudin and John Grundy. 2011. Generating essential user interface prototypes to validate requirements. In *2011 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011)*. 564–567. https://doi.org/10.1109/ASE.2011.6100126

[34] Sandra Kelly. 2010. Towards an evolutionary framework for agile requirements elicitation. In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Berlin, Germany) *(EICS '10)*. Association for Computing Machinery, New York, NY, USA, 349–352. https://doi.org/10.1145/1822018.1822078

[35] Suzanne Kieffer and Jean Vanderdonckt. 2023. A Comparison of Paper Sketch and Interactive Wireframe by Eye Movements Analysis, Survey, and Interview. In *Proceedings of 56th Hawaii International Conference on System Sciences* (Maui, Hawaii, USA) *(HICSS '23)*, Tung X. Bui (Ed.). ScholarSpace, 4067–4076. https://hdl.handle.net/10125/103128

[36] Hyung Nam Kim, Tonya L. Smith-Jackson, and Chang S. Nam. 2013. Elicitation of Haptic User Interface Needs of People with Low Vision. *Int. J. Hum. Comput. Interact.* 29, 7 (2013), 488–500. https://doi.org/10.1080/10447318.2012.722465

[37] James A. Landay and Brad A. Myers. 2001. Sketching Interfaces: Toward More Human Interface Design. *Computer* 34, 3 (2001), 56–64. https://doi.org/10.1109/2.910894

[38] Judith M. Leson, Valbjörn Jón Valbjörnsson, Matthias Book, and Marc Hesenius. 2026. Say It or Sketch It: Cross-Group Elicitation of Voice and Sketch Interactions. In *Proceedings of the Extended Abstracts of the 2026 CHI Conference on Human Factors in Computing Systems, CHI EA 2026, Barcelona, Spain, April 13-17, 2026*, Nuria Oliver, David A. Shamma, Heloisa Candello, Pablo César, Pedro Lopes, Valentino Artizzu, Fiona Draxler, Gustavo López, Anke V. Reinschluessel, Xin Tong, and Phoebe O. Toups Dugas (Eds.). Association for Computing Machinery, 576:1–576:6. https://doi.org/10.1145/3772363.3799271

[39] Chunlei Li, Chris A. McMahon, and Linda B. Newnes. 2009. Annotation in Design Processes: Classification of Approaches. In *Proceedings of the 17th International Conference on Engineering Design* (Stanford, CA, USA) *(ICED '09, Vol. 8)*, M. Norell Bergendahl, M. Grimheden, L. Leifer, P. Skogstad, and U. Lindemann (Eds.). Design Information and Knowledge, Glasgow, Scotland, 251–262. https://www.designsociety.org/publication/28818/annotation_in_design_processes_classification_of_approaches

[40] Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of Psychology* 22, 140 (1932), 55–. http://psycnet.apa.org/record/1933-01885-001

[41] James Lin, Mark W. Newman, Jason I. Hong, and James A. Landay. 2001. DENIM: an informal tool for early stage web site design. In *CHI 2001 Extended Abstracts on Human Factors in Computing Systems, CHI Extended Abstracts 2001, Seattle, Washington, USA, March 31 - April 5, 2001*, Marilyn Mantai Tremaine (Ed.). Association for Computing Machinery, 205–206. https://doi.org/10.1145/634067.634190

[42] Grzegorz Loniewski, Emilio Insfran, and Silvia Abrahão. 2010. A Systematic Review of the Use of Requirements Engineering Techniques in Model-Driven Development. In *Model Driven Engineering Languages and Systems*, Dorina C. Petriu, Nicolas Rouquette, and Øystein Haugen (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 213–227.

[43] Nathan Magrofuoco, Paolo Roselli, and Jean Vanderdonckt. 2021. Two-dimensional Stroke Gesture Recognition: A Survey. *ACM Comput. Surv.* 54, 7, Article 155 (July 2021), 36 pages. https://doi.org/10.1145/3465400

[44] Nathan Magrofuoco, Arthur Sluÿters, Paolo Roselli, and Jean Vanderdonckt. 2025. Comparative Testing of 2D Stroke Gesture Recognizers in Multiple Contexts. *Proc. ACM Hum.-Comput. Interact.* 9, 4, Article EICS005 (June 2025), 36 pages. https://doi.org/10.1145/3733051

[45] Nathan Magrofuoco and Jean Vanderdonckt. 2019. Gelicit: A Cloud Platform for Distributed Gesture Elicitation Studies. *Proc. ACM Hum.-Comput. Interact.* 3, EICS, Article 6 (jun 2019), 41 pages. https://doi.org/10.1145/3331148

[46] Martin C. Maguire and Nigel Bevan. 2002. User Requirements Analysis: A Review of Supporting Methods. In *Usability: Gaining a Competitive Edge, IFIP 17th World Computer Congress - TC13 Stream on Usability: Gaining a Competitive Edge, August 25-29, 2002, Montréal, Québec, Canada (IFIP Conference Proceedings, Vol. 226)*, Judy Hammond, Tom Gross, and Janet Wesson (Eds.). Kluwer, 133–148.

[47] Abel Menkveld, Sjaak Brinkkemper, and Fabiano Dalpiaz. 2019. User Story Writing in Crowd Requirements Engineering: The Case of a Web Application for Sports Tournament Planning. In *Proceedings of the 27th IEEE International Requirements Engineering Conference Workshops (REW)*. 174–179. https://doi.org/10.1109/REW.2019.00037

[48] Meredith Ringel Morris, Andreea Danielescu, Steven Mark Drucker, Danyel Fisher, Bongshin Lee, m. c. schraefel, and Jacob O. Wobbrock. 2014. Reducing legacy bias in gesture elicitation studies. *Interactions* 21, 3 (2014), 40–45. https://doi.org/10.1145/2591689

[49] Yeshica Isela Ormeño, José Ignacio Panach, and Oscar Pastor. 2023. An empirical experiment of a usability requirements elicitation method to design GUIs based on interviews. *Information and Software Technology* 164 (2023), 107324. https://doi.org/10.1016/j.infsof.2023.107324

[50] Ilia A. Ovsiannikov, Michael A. Arbib, and Thomas H. McNeill. 1999. Annotation technology. *International Journal of Human-Computer Studies* 50, 4 (1999), 329–362. https://doi.org/10.1006/ijhc.1999.0247

[51] Samuli Pekkola, Niina Kaarilahti, and Pasi Pohjola. 2006. Towards formalised end-user participation in information systems development process: bridging the gap between participatory design and ISD methodologies. In *Proceedings of the Ninth Conference on Participatory Design: Expanding Boundaries in Design - Volume 1* (Trento, Italy) *(PDC '06)*. Association for Computing Machinery, New York, NY, USA, 21–30. https://doi.org/10.1145/1147261.1147265

[52] Chris Porter, Emmanuel Letier, and M. Angela Sasse. 2014. Building a National E-Service using Sentire experience report on the use of Sentire: A volere-based requirements framework driven by calibrated personas and simulated user feedback. In *Proc. of 22nd IEEE International Requirements Engineering Conference* (Karlskrona, Sweden) *(RE '14)*. IEEE Computer Society Press, Los Alamitos, US, 374–383. https://doi.org/10.1109/RE.2014.6912288

[53] José Matías Rivero, Gustavo Rossi, Julián Grigera, Juan Burella, Esteban Robles Luna, and Silvia Gordillo. 2010. From Mockups to User Interface Models: An Extensible Model Driven Approach. In *Current Trends in Web Engineering*, Florian Daniel and Federico Michele Facca (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 13–24.

[54] Hugo Romat, Emmanuel Pietriga, Nathalie Henry-Riche, Ken Hinckley, and Caroline Appert. 2019. SpaceInk: Making Space for In-Context Annotations. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 871–882. https://doi.org/10.1145/3332165.3347934

[55] Ugo Braga Sangiorgi, François Beuvens, and Jean Vanderdonckt. 2012. User interface design by collaborative sketching. In *Designing Interactive Systems Conference 2012, DIS '12, Newcastle Upon Tyne, United Kingdom, June 11-15, 2012*. Association for Computing Machinery, 378–387. https://doi.org/10.1145/2317956.2318013

[56] Martin Schrepp, Andreas Hinderks, and Jörg Thomaschewski. 2017. Construction of a Benchmark for the User Experience Questionnaire (UEQ). *Int. J. Interact. Multim. Artif. Intell.* 4, 4 (2017), 40–44. https://doi.org/10.9781/ijimai.2017.445

[57] Martin Schrepp and Jörg Thomaschewski. 2019. Design and Validation of a Framework for the Creation of User Experience Questionnaires. *International Journal of Interactive Multimedia and Artificial Intelligence* 5, 7 (2019), 88–95. https://doi.org/10.9781/IJIMAI.2019.06.006

[58] Vinícius C. V. B. Segura, Simone D. J. Barbosa, and Fabiana Pedreira Simões. 2012. UISKEI: a sketch-based prototyping tool for defining and evaluating user interface behavior. In *Proceedings of the International Working Conference on Advanced Visual Interfaces* (Capri Island, Italy) *(AVI '12)*. Association for Computing Machinery, New York, NY, USA, 18–25. https://doi.org/10.1145/2254556.2254564

[59] Azita Soltanian Bojnord, Rodina Binti Ahmad, and Hoda Soltanian Bojnord. 2013. WEBSTUIRE: WEB-based Support Tool for user interface requirements elicitation. In *Proceedings of International eConference on Computer and Knowledge Engineering* (Mashhad, Iran) *(ICCKE 2013)*. IEEE Computer Society Press, Los Alamitos, USA, 52–58. https://doi.org/10.1109/ICCKE.2013.6682840

[60] Kênia Sousa, Hildeberto Mendonça, Jean Vanderdonckt, Els Rogier, and Joannes Vandermeulen. 2008. User Interface Derivation from Business Processes: A Model-Driven Approach for Organizational Engineering. In *Proceedings of the ACM Symposium on Applied Computing* (Fortaleza, Ceara, Brazil) *(SAC '08)*. Association for Computing Machinery, New York, NY, USA, 553–560. https://doi.org/10.1145/1363686.1363821

[61] Maximilian Speicher and Michael Nebeling. 2018. GestureWiz: A Human-Powered Gesture Design Environment for User Interface Prototypes. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3173574.3173681

[62] Tung-Tien Sun. 2020. Active versus passive reading: how to read scientific papers? *National Science Review* 7, 9 (06 2020), 1422–1427. https://doi.org/10.1093/nsr/nwaa130 arXiv:https://academic.oup.com/nsr/article-pdf/7/9/1422/38881853/nwaa130.pdf

[63] Craig J. Sutherland, Andrew Luxton-Reilly, and Beryl Plimmer. 2016. Freeform digital ink annotations in electronic documents: A systematic mapping study. *Computers & Graphics* 55 (2016), 1–20. https://doi.org/10.1016/j.cag.2015.10.014

[64] Maroua Tikat, Aline Menin, Michel Buffa, and Marco Winckler. 2022. Engineering Annotations to Support Analytical Provenance in Visual Exploration Processes. In *Web Engineering - 22nd International Conference, ICWE 2022, Bari, Italy, July 5-8, 2022, Proceedings (Lecture Notes in Computer Science, Vol. 13362)*, Tommaso Di Noia, In-Young Ko, Markus Schedl, and Carmelo Ardito (Eds.). Springer, 204–218. https://doi.org/10.1007/978-3-031-09917-5_14

[65] Matias Urbieta, Nahime Torres, José Matias Rivero, Gustavo Rossi, and F. J. Dominguez-Mayo. 2018. Improving Mockup-Based Requirement Specification with End-User Annotations. In *Agile Processes in Software Engineering and Extreme Programming*, Juan Garbajosa, Xiaofeng Wang, and Ademar Aguiar (Eds.). Springer International Publishing, Cham, 19–34.

[66] Radu-Daniel Vatavu. 2017. Improving Gesture Recognition Accuracy on Touch Screens for Users with Low Vision. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, Denver, CO, USA, May 06-11, 2017*, Gloria Mark, Susan R. Fussell, Cliff Lampe, m. c. schraefel, Juan Pablo Hourcade, Caroline Appert, and Daniel Wigdor (Eds.). Association for Computing Machinery, 4667–4679. https://doi.org/10.1145/3025453.3025941

[67] Radu-Daniel Vatavu. 2019. The Dissimilarity-Consensus Approach to Agreement Analysis in Gesture Elicitation Studies. In *Proceedings of the ACM Int. Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, Stephen A. Brewster, Geraldine Fitzpatrick, Anna L. Cox, and Vassilis Kostakos (Eds.). Association for Computing Machinery, New York, NY, USA, 224. https://doi.org/10.1145/3290605.3300454

[68] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. 2012. Gestures as point clouds: a $P recognizer for user interface prototypes. In *International Conference on Multimodal Interaction, ICMI '12, Santa Monica, CA, USA, October 22-26, 2012*, Louis-Philippe Morency, Dan Bohus, Hamid K. Aghajan, Justine Cassell, Anton Nijholt, and Julien Epps (Eds.). Association for Computing Machinery, 273–280. https://doi.org/10.1145/2388676.2388732

[69] Radu-Daniel Vatavu and Jacob O. Wobbrock. 2015. Formalizing Agreement Analysis for Elicitation Studies: New Measures, Significance Test, and Toolkit. In *Proceedings of the ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 1325–1334. https://doi.org/10.1145/2702123.2702223

[70] Santiago Villarreal-Narvaez, Arthur Sluÿters, Jean Vanderdonckt, and Radu-Daniel Vatavu. 2024. Brave New GES World: A Systematic Literature Review of Gestures and Referents in Gesture Elicitation Studies. *ACM Comput. Surv.* 56, 5, Article 128 (jan 2024), 55 pages. https://doi.org/10.1145/3636458

[71] Santiago Villarreal-Narvaez, Jean Vanderdonckt, Radu-Daniel Vatavu, and Jacob O. Wobbrock. 2020. A Systematic Review of Gesture Elicitation Studies: What Can We Learn from 216 Studies?. In *Proceedings of the ACM Int. Conf. on Designing Interactive Systems Conference* (Eindhoven, Netherlands) *(DIS '20)*. Association for Computing Machinery, New York, NY, USA, 855–872. https://doi.org/10.1145/

3357236.3395511

[72] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-Defined Gestures for Surface Computing. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '09)*. Association for Computing Machinery, New York, NY, USA, 1083–1092. https://doi.org/10.1145/1518701.1518866

[73] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. 2007. Gestures without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA) *(UIST '07)*. Association for Computing Machinery, New York, NY, USA, 159–168. https://doi.org/10.1145/1294211.1294238

[74] Haijun Xia, Michael Glueck, Michelle Annett, Michael Wang, and Daniel Wigdor. 2022. Iteratively Designing Gesture Vocabularies: A Survey and Analysis of Best Practices in the HCI Literature. *ACM Trans. Comput. Hum. Interact.* 29, 4 (2022), 37:1–37:54. https://doi.org/10.1145/3503537

[75] Seongjun Yun, Minseok Pang, Hongjin Cho, Jongho Chae, Yoonjung Choi, and Eun-Seok Lee. 1999. User-friendly support environment for requirement analysis in user interface design. In *Proceedings of the 1999 ICPP Workshops on Collaboration and Mobile Computing (CMC'99). Group Communications (IWGC). Internet '99 (IWI'99). Industrial Applications on Network Computing (INDAP). Multime*. IEEE Computer Society Press, Los Alamitos, USA, 414–417. https://doi.org/10.1109/ICPPW.1999.800094

[76] Ionut-Alexandru Zaiti, Stefan-Gheorghe Pentiuc, and Radu-Daniel Vatavu. 2015. On Free-Hand TV Control: Experimental Results on User-Elicited Gestures with Leap Motion. *Personal and Ubiquitous Computing* 19, 5-6 (2015), 821–838. https://doi.org/10.1007/s00779-015-0863-y

[77] Shumin Zhai, Per Ola Kristensson, Caroline Appert, Tue Haste Anderson, and Xiang Cao. 2012. Foundational Issues in Touch-Surface Stroke Gesture Design — An Integrative Review. *Foundations and Trends in Human–Computer Interaction* 5, 2 (2012), 97–205. https://doi.org/10.1561/1100000012

[78] Michał Śmiałek and Tomasz Straszak. 2012. Facilitating transition from requirements to code with the ReDSeeDS tool. In *Prcoeedings of the 20th IEEE International Requirements Engineering Conference* (Chicago, IL, USA) *(RE '12)*. IEEE Computer Society Press, Los Alamitos, USA, 321–322. https://doi.org/10.1109/RE.2012.6345825

A Activity Diagrams of Crowditivity

This section reproduces the Activity Diagrams in UML V2.5 notation that resulted from the focus group (Fig. 1-d).

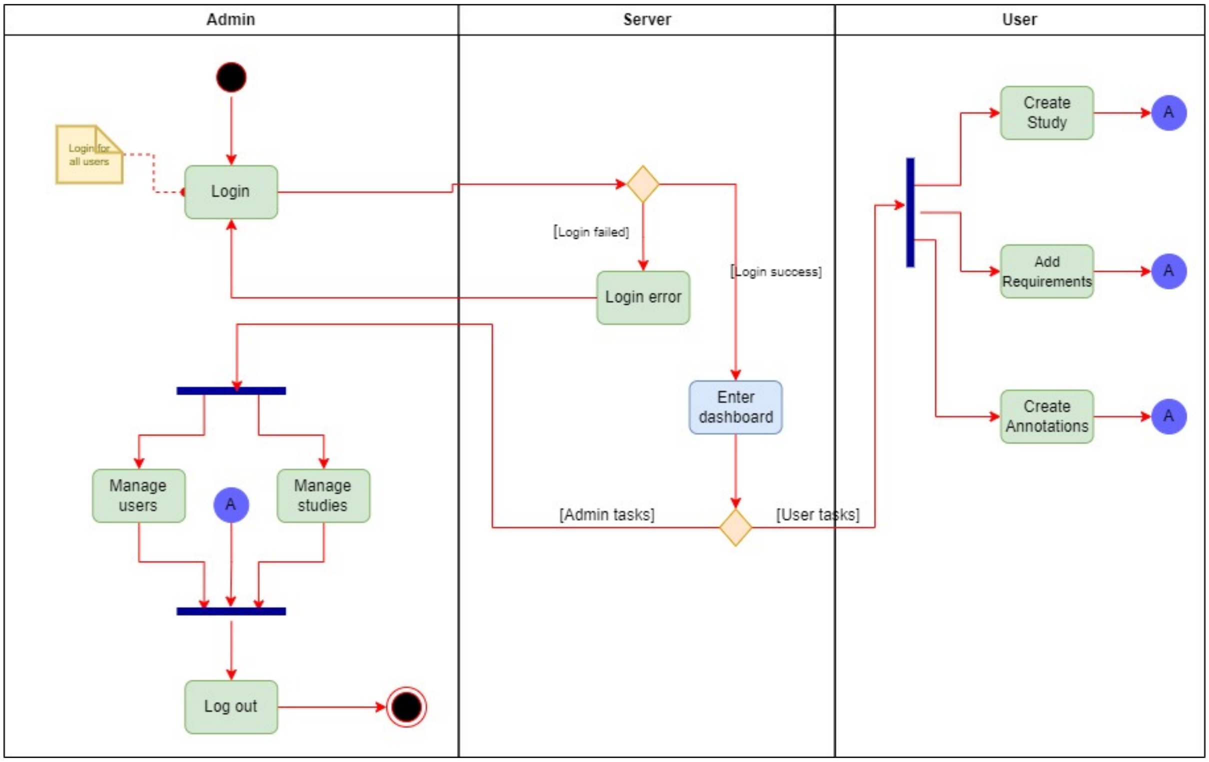


Fig. 16. UML Main Activity Diagram of Crowditivity.

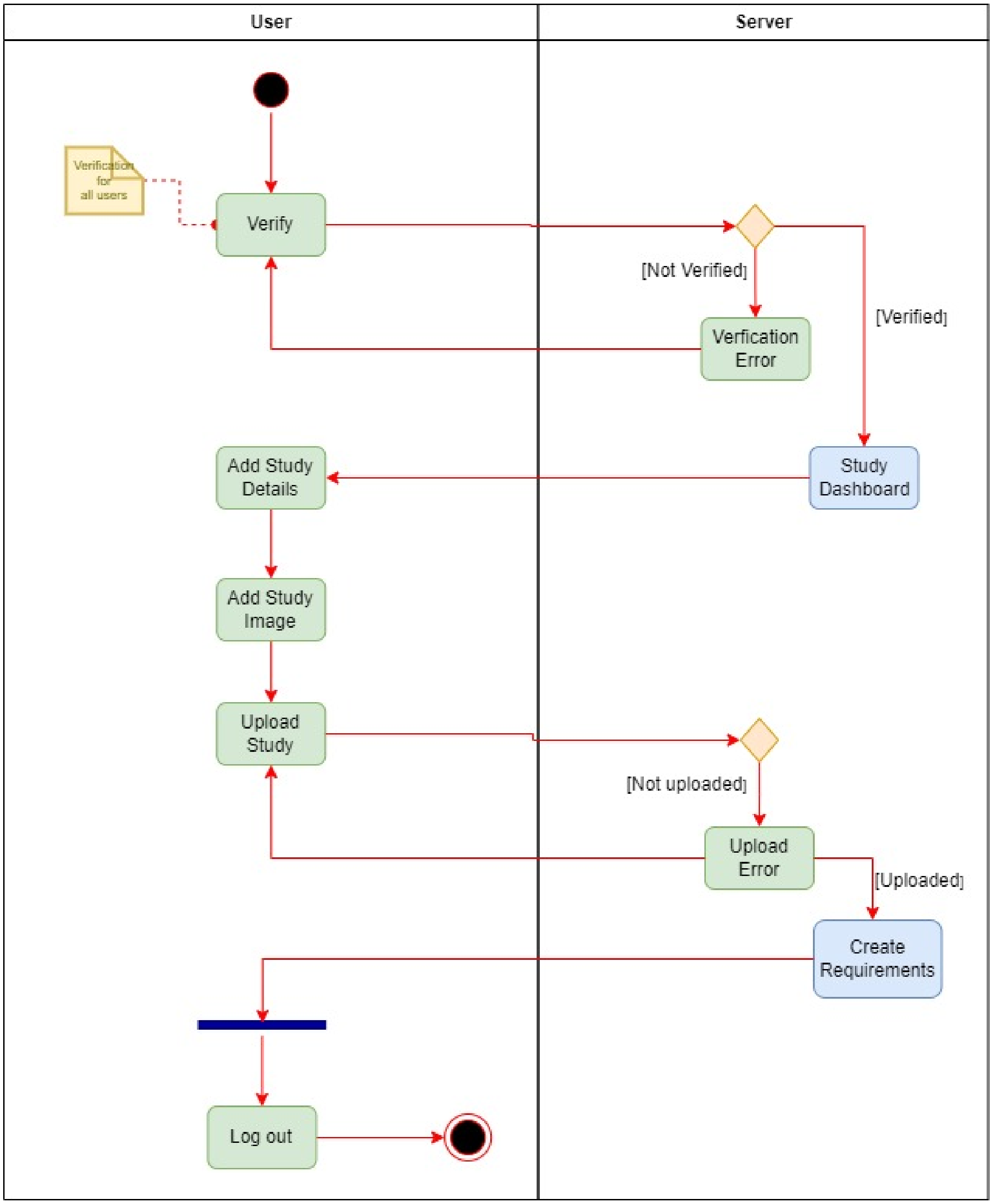


Fig. 17. UML Activity Diagram of Crowditivity for "Create a Study".

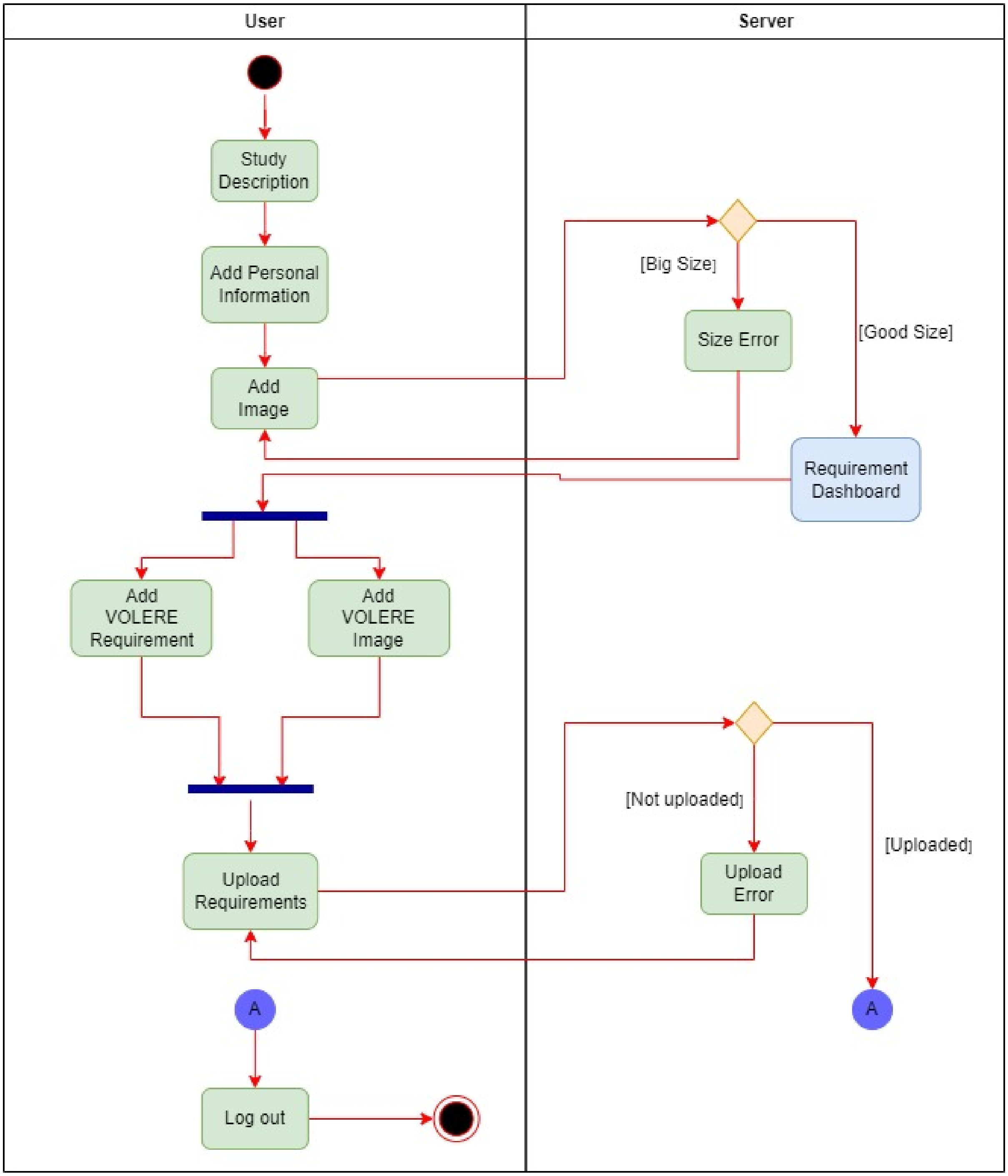


Fig. 18. UML Activity Diagram of Crowditivity for "Create a Requirement".

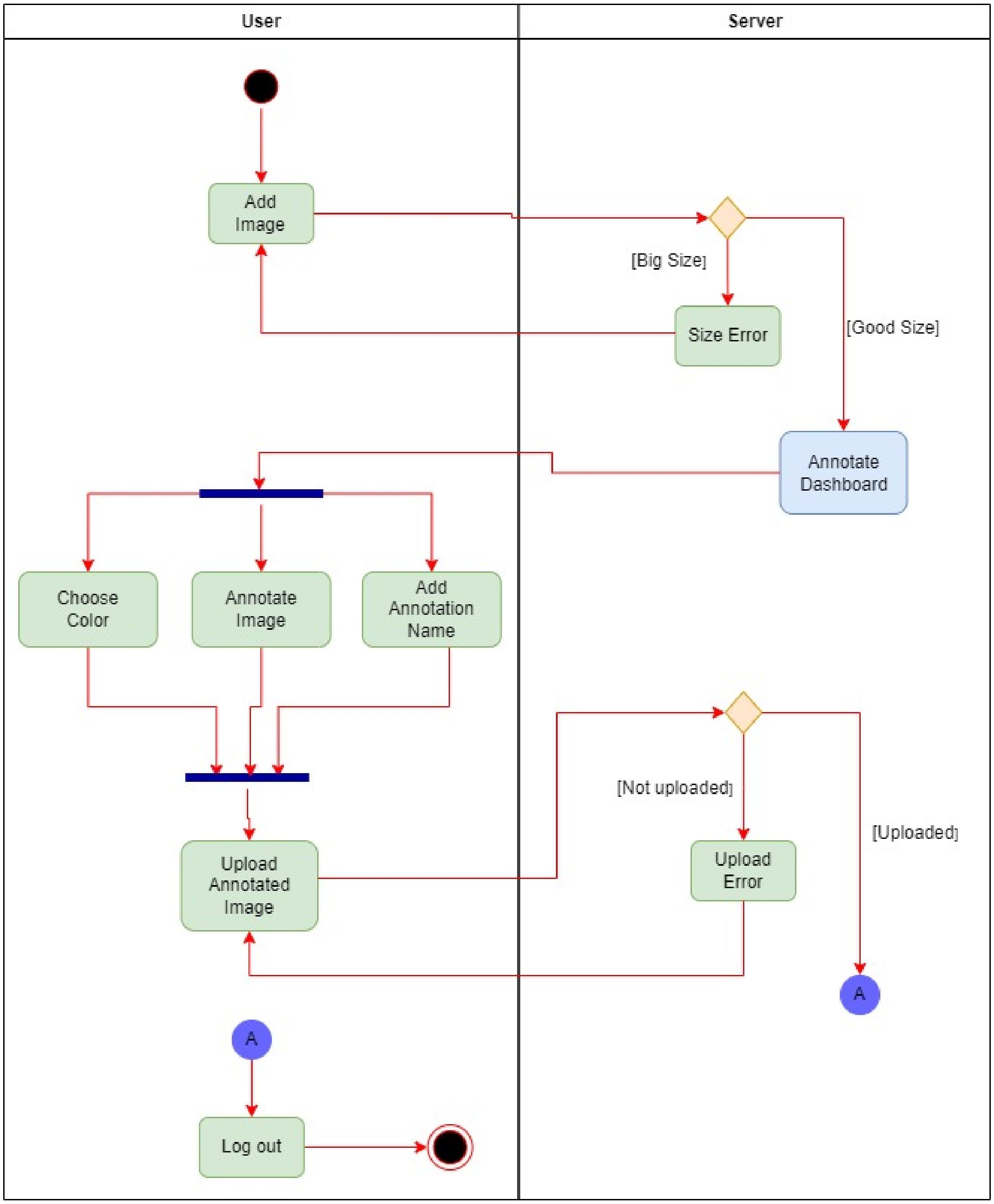


Fig. 19. UML Activity Diagram of Crowditivity for "Add an annotation".

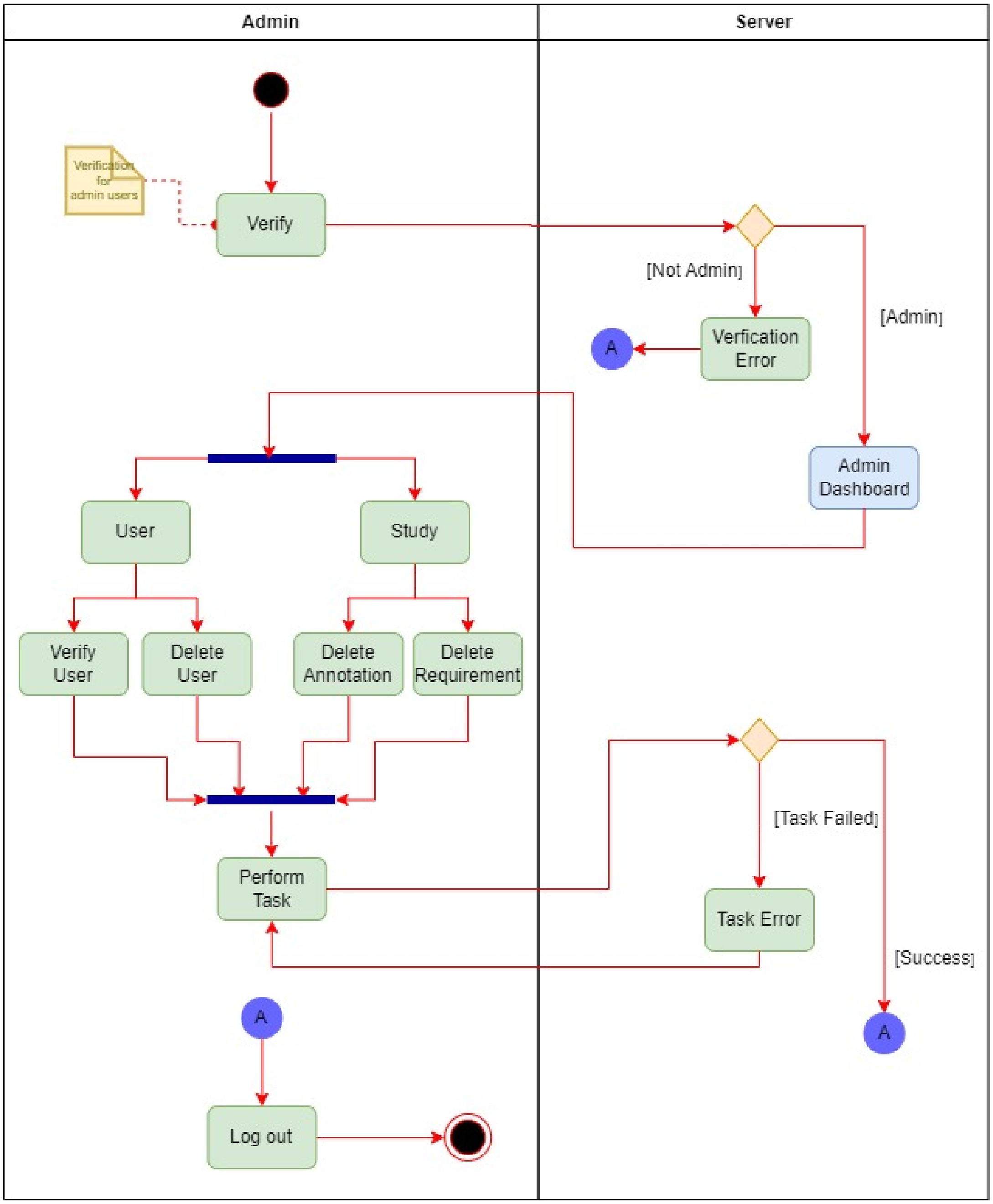


Fig. 20. UML Activity Diagram of Crowditivity for administrator.

B Practicing of the Annotation Types

This section reproduces the user interfaces developed for participants to practice the gestures associated with each of the 18 annotation types and see their recognition, before starting the user study.

Fig. 21. Gesture for "Add".

Fig. 22. Gesture for "Attention".

Fig. 23. Gesture for "Bad".

Fig. 24. Gesture for "Confusion".

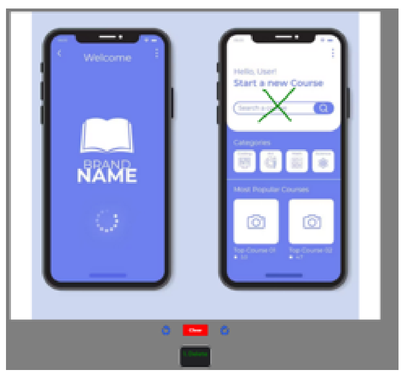


Fig. 25. Gesture for "Delete".

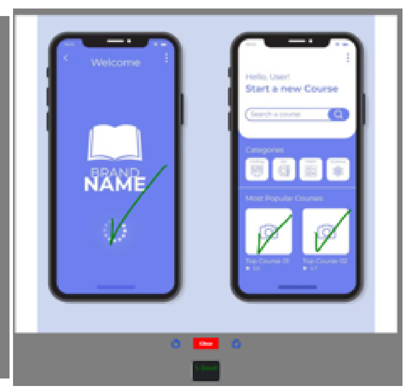


Fig. 26. Gesture for "Good".

Fig. 27. Gesture for "Line".

Fig. 28. Gesture for "Move down".

Fig. 29. Gesture for "Move up".

Fig. 30. Gesture for "Move left".

Fig. 31. Gesture for "Move right".

Fig. 32. Gesture for "Select".

Fig. 33. Gesture for "Rotate left".

Fig. 34. Gesture for "Rotate right".

Fig. 35. Gesture for "Optional".

Fig. 36. Gesture for "Required".

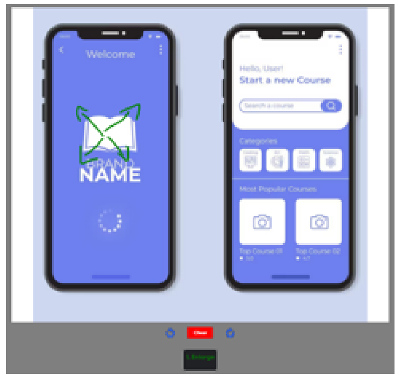


Fig. 37. Gesture for "Zoom in".

Fig. 38. Gesture for "Zoom out".

B.1 UML V2.5 Class Diagram of Crowditivity

Fig. 39 shows a UML V2.5 Class Diagram that was designed to implement the database of Crowditivity: any user can create as many studies as desired, for which any set of UI requirements can be entered in Volere or a generic format. Any UI requirement can be attached to zero to many UI elements. Any UI element can be the subject of an unlimited number of annotations that are related to regions of the mockup or UI element in question. Any annotation is created by any user, not necessarily the person who entered the requirement or the UI element. In this way, crowdsourcing is ensured by enabling stakeholders to edit any user study, requirement, UI element, and annotation in any order while being distributed in time and space, as recommended by Ali et al. [1, 2].

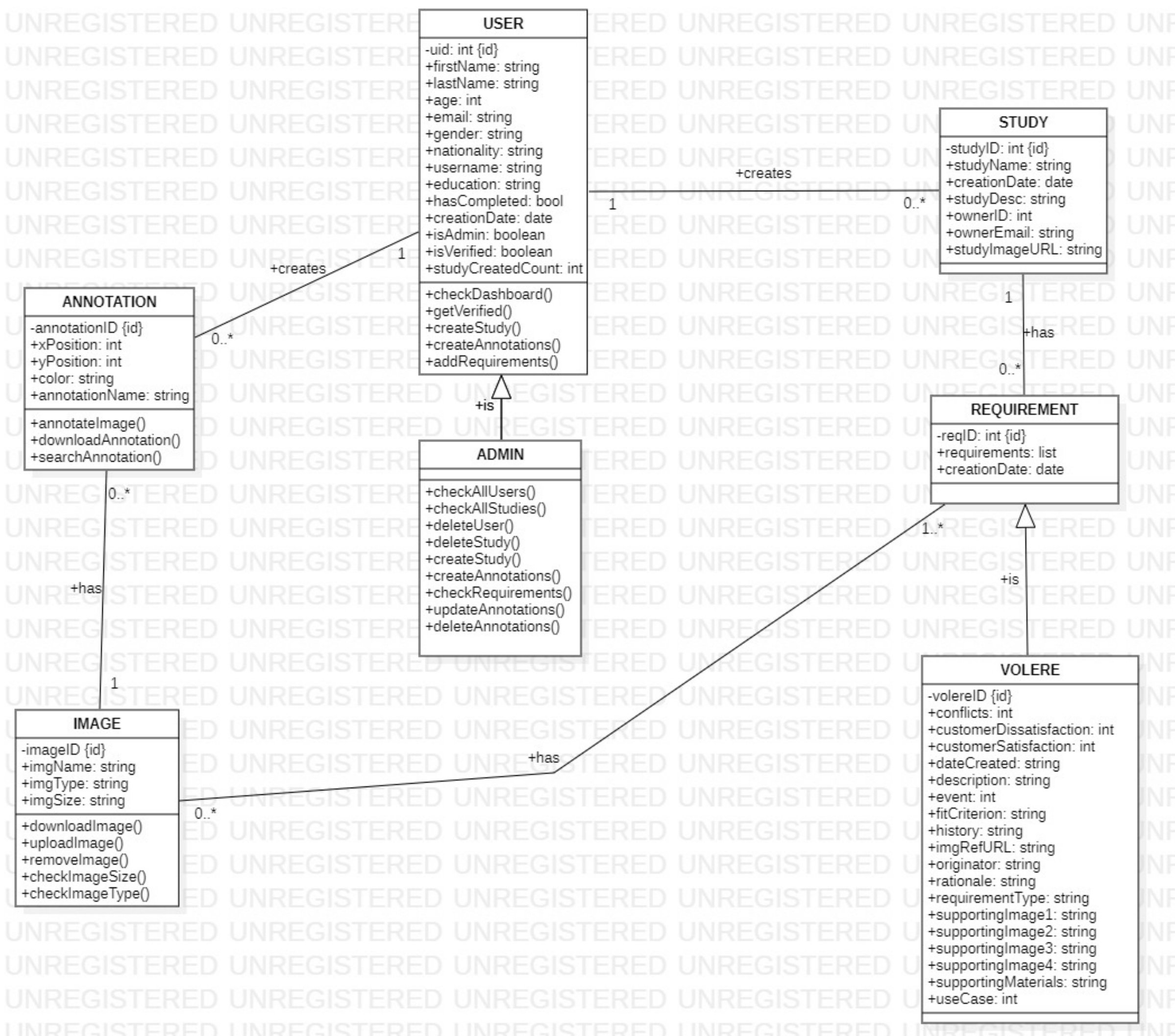


Fig. 39. UML V2.5 Class Diagram of Crowditivity.